TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

SAMI ERKKILÄ

ALGORITMIAVUSTEISEN SUUNNITTELUN HYÖDYNTÄMINEN BETONIELEMENTTIRAKENTEIDEN SUUNNITTELUSSA

Diplomityö

Tarkastaja:
professori Mikko Malaska
Tarkastaja ja aihe hyväksytty Talouden ja rakentamisen tiedekuntaneuvoston kokouksessa 27. maaliskuuta 2017

TIIVISTELMÄ

SAMI ERKKILÄ: Algoritmiavusteisen suunnittelun hyödyntäminen betonielementtirakenteiden suunnittelussa
Tampereen teknillinen yliopisto
Diplomityö, 98 sivua, 5 liitesivua
Toukokuu 2017
Rakennustekniikan diplomi-insinöörin tutkinto-ohjelma
Pääaine: Rakennesuunnittelu
Tarkastaja: professori Mikko Malaska

Avainsanat: algoritmi, algoritmiavusteinen suunnittelu, parametri, elementtisuunnittelu, tietomalli, BIM, Grasshopper

Algoritmiavusteisessa suunnittelussa pyritään hyödyntämään algoritmeja, jotka suorittavat tietyn tehtäväsarjan. Algoritmi on ennalta määrätty ja yksiselitteinen tehtäväsarja, jonka jokainen osa on tarkkaan määrätty ja niillä on tietty tehtävä osana sarjaa. Tässä tutkimuksessa pyritään selvittämään, voidaanko algoritmiavusteista suunnittelua hyödyntää betonielementtirakenteiden suunnittelussa.

Tutkimuksen päätavoitteina on luoda betonisten väliseinäelementtien algoritmiavusteinen suunnitteluprosessi, selvittää millä tavalla algoritmiavusteinen suunnittelu vaikuttaa betonielementtirakenteiden suunnitteluun ja minkälaisia ongelmia ja haasteita on tämän hetken ohjelmistoissa. Tutkimuksen tavoitteiden saavuttamiseksi hyödynnettiin kahta tutkimusmenetelmää: kirjallisuusselvitystä ja case-tutkimusta. Kirjallisuusselvityksen avulla kartoitettiin tutkimuksen eri alueiden taustoja, kuten algoritmin luomista ja sen jäsentelyä. Sen avulla saatiin selville myös, että globaalisti rakennusalalla algoritmiavusteisia suunnittelumenetelmiä hyödynnetään laaja-alaisesti. Näiden lisäksi luotiin betonisten väliseinäelementtien algoritmiavusteinen suunnitteluprosessi.

Case-tutkimuksessa testattiin suunnitteluprosessin toimivuutta toteutetun kerrostalokohteen väliseinäelementtien suunnitteluun. Tutkimuksessa käytettiin Rhinoceros 3D, Grasshopper ja Tekla Structures -ohjelmistoja. Tutkimuksen aikana ilmeni useita ongelmia suunnitteluprosessissa ja ohjelmistoissa. Ohjelmistojen välinen yhteys aiheutti useita hankaluuksia käytettävyydelle ja ohjelmien toimivuudelle. Väliseinien mallintaminen onnistui tehokkaasti, mutta väliseinien elementtijaon algoritmiavusteista suunnittelua ei saatu toteutettua elementtijaon suunnittelun monimutkaisuuden vuoksi. Lisäksi väliseinien oviaukkojen ja muiden detaljointien kanssa todettiin ongelmia.

Tulosten analysoinnissa havaittiin, että algoritmiavusteinen elementtisuunnittelu tulisi tehdä kerroksittain. Löydettiin myös keinoja, joiden avulla osa prosessin ongelmista voitaisiin ratkaista. Analysoinnin lopputuloksena todettiin, että algoritmiavusteisessa suunnittelussa on paljon potentiaalia, mutta suunnitteluprosessia tulee kuitenkin vielä muokata. Koska prosessi ei sovellu suoraan käytettäväksi, on tutkittava ja valittava halutaanko hyödyntää ainoastaan tehokasta ja eksaktia mallintamista, vai pyritäänkö ratkaisemaan nykyiset ongelmat ja mahdollistamaan myös algoritmiavusteinen detaljointi. Tämän tutkimuksen tulokset antavat hyvän taustan ja lähtökohdan jatkotutkimuksella. Tutkimustulosten perusteella voidaankin todeta algoritmiavusteinen suunnittelun olevan tämän hetken ja tulevaisuuden suunnittelijoiden työkalu.

ABSTRACT

SAMI ERKKILÄ: Utilization of algorithm-aided design in the design process of precast concrete elements
Tampere University of Technology
Master of Science Thesis, 98 pages, 5 Appendix pages
May 2017
Master's Degree Programme in Civil Engineering
Major: Structural Engineering
Examiner: Professor Mikko Malaska

Keywords: algorithm, algorithm-aided design, parameter, precast concrete design, building information model, BIM, Grasshopper

In algorithm-aided design there is a strive to utilize algorithms, which are used to do a specific series of tasks. An algorithm is a predefined and conclusive series of tasks, in which every part of it is precisely definite and have a specific role as a part of the series. In this Master's thesis research report, the goal is to study if there is a way to utilize algorithm-aided design in the design process of precast concrete element.

The goals of this study are to create an algorithm-aided design process of precast concrete partition walls, clarify the potential of algorithm-aided design in precast concrete design process, and investigate what kind of problems or difficulties there is in the programs used in the process. To accomplish the goals two different research methods were used: literature survey and case study. The frame of reference of the research context was written using the latest national and international literature. The review demonstrated that parametric design techniques and the algorithm-aided design have been used in a wide range of applications. Using the literature references, an algorithm-aided design process was developed for precast concrete partition walls.

The usability of the design process developed was studied using a case study. Design software used in the case study were Rhinoceros 3D, Grasshopper, and Tekla Structures. Several problems in the design process and within the software were encountered during the case study. The links between the three software were causing complicated problems for the usability and the functionality of the software. Although the algorithm-aided modelling was successful and efficient, dividing the walls into wall-elements was causing problems because the element-formation process is rather complicated. Additionally, there were couple of problems regarding openings and detailing of the walls.

The case study demonstrated that the algorithm-aided precast concrete design should be done storey by storey. Some of the problems encountered in the case study were solved during the design process analysis. The result of the analysis was that there is a lot of potential for the algorithm-aided design, but the design process created needs some modification to be utilized properly. There were two further development paths, one was to solely utilize the efficient and exact algorithm-aided modelling, and the other was to try to solve the problems and fully utilize the algorithm-aided design for the whole precast concrete element design process. The results of this Master thesis research provide good bases for the further development. To conclude, based on the results of this study, algorithm-aided design is now and in the future an efficient tool for designers.

ALKUSANAT

Tämä työ on tehty Tampereen teknillisessä yliopistossa. Kiitos kaikille henkilöille, jotka ovat auttaneet ja tsempanneet minua tämän kevään aikana.

Iso kiitos Ramboll Finland Oy:lle ja Inari Weijolle, jotka mahdollistivat tämän diplomityön toteuttamisen. Rambollilta haluan kiittää myös ohjaajiani Markku Raiskilaa, Max Levanderia ja kaikkia muitakin ohjauksessa mukana olleita. Kiitos muillekin työkavereille, jotka kuuntelivat, ohjasivat, neuvoivat ja tsemppasivat! Kiitos myös Tampereen teknillisen yliopiston ohjaajilleni Mikko Malaskalle ja Toni Teittiselle.

Kuitenkin suurin kiitos kuuluu kihlatulleni, Hennille, joka on koko tutkintoni ajan ollut suurin inspiraationlähteeni ja tukeni. Kiitos myös kaikille ystävilleni yliopistolla, varsinkin Raksajatuille loistavista viidestä teekkarivuodesta.

The Road goes ever on and on

Out from the door where it began.

Now far ahead the Road has gone.

Let others follow, if they can!

Let them a journey new begin.

But I at last with weary feet

Will turn towards the lighted inn,

My evening-rest and sleep to meet.

— J.R.R. Tolkien, The Lord of the Rings

Tampereella, 21.05.2017

Sami Erkkilä

SISÄLLYSLUETTELO

KUVALUETTELO

LYHENTEET JA MERKINNÄT

AAB	Algorithm-Aided Building Information Modeling (Humppi 2015) Algoritmiavusteinen tietomallintaminen tai algoritmiavusteinen tietomalli
AAD	Algorithm-Aided Design (Humppi 2015, Tanska, Österlund 2014) Algoritmiavusteinen suunnittelu
BIM	Building Information Modeling, eli tietomallintaminen
CAD	Computer-Aided design, tietokoneavusteinen suunnittelu
Rhino	Rhinoceros 3D -mallinnus ohjelmisto
Solibri	Solibri Model Checker -ohjelmisto, tietomallien tarkastusohjelmisto
Tekla	Tekla Structures -tietomallinnus ohjelmisto

KÄSITTEET

Algoritmi	Algoritmi on tehtäväsarja, jolla on alku ja loppu. Tehtäväsarjan jokainen osa tulee olla tarkasti määrätty ja ne ovat olennainen osa kokonaisuutta. Algoritmi voidaan luoda tekstipohjaisella koodilla tai visuaalisella koodilla.
Algoritmiavusteinen suunnittelu	Suunnitteluprosessi, joka hyödyntää algoritmeja. Samalla tavalla kuin tietomallit, ovat algoritmitkin suunnittelutyön työkaluja ja apuvälineitä.
Algoritmiavusteinen mallintaminen	Algoritmiavusteisen suunnittelun yksi sovelluskohteista. Algoritmilla ohjattavaa tietomallintamista. Esimerkiksi Grasshopperilla voidaan algoritmeilla ohjata Tekla Structuresia.
Parametri	Parametri on muuttuja tai lähtöarvo. Yleensä se on jokin numeroarvo, kuten kerrosten lukumäärä tai kerroskorkeus.
Suunnitteluprosessi	Suunnitteluprosessi on jokin määrätty ja rajallinen tehtäväsarja. Sillä selitetään tehtäväsarjan jokainen vaihe ja niiden järjestys. Yleensä suunnitteluprosessin havainnollistamiseksi se esitetään kaavion muodossa.
Tietomalli	Tietomalli sisältää rakennusosiin liittyvää geometriaa ja valmistustietoa kuten materiaalitietoja. Mallin sisältämä tieto pitää olla kolmiulotteista, digitaalista, ymmärrettävää ja hyödynnettävissä olevaa.
Visuaalinen skriptaus	Visuaalinen skriptaus on ohjelmointia, jossa hyödynnetään valmiiksi ohjelmoituja komponentteja. Komponentteja yhdistelemällä luodaan algoritmeja ja algoritmeja hyödyntämällä suunnittelutyössä puhutaan algoritmiavusteisesta suunnittelusta.

1. JOHDANTO

Building Information Modeling (BIM) eli tietomallintaminen on viimeisten kymmenien vuosien ajan vauhdittanut rakennesuunnittelun kehittymistä. Tietomallintamista ennen rakennusalan suunnittelua hallitsi Computer Aided Desing (CAD) ja vielä kauempana historiassa alaa hallitsi piirtopöydät. Jokaisesta vaiheesta seuraavaan siirryttäessä tapa suunnitella muuttui. Uudet työkalut ja suunnittelutavat mahdollistivat suunnittelijoiden ajankäytön tehostamisen parempien laitteiden, ohjelmistojen ja käytäntöjen avulla. Algoritmiavusteinen suunnittelu tulee olemaan seuraava askel eteenpäin.

1.1 Tutkimuksen tausta ja motivaatio

Algoritmiavusteinen suunnittelu tulee muuttamaan tapaamme suunnitella rakennuksia ja rakenteita. Elementtisuunnittelu on vahvasti sidonnainen tietomallintamiseen tällä hetkellä, joten tässä työssä tutkitaan myös paljon algoritmiavusteista mallintamista. Algoritmiavusteinen mallintaminen on algoritmiavusteista suunnittelua ja parametrista mallintamista. Parametrisuus voidaan ajatella riippuvuussuhteena, jossa esimerkiksi pilarin geometria määrittää siihen liittyvän pilarikonsolin koon, pilarikonsolin koko taas määrittää siihen liittyvän palkin geometriaa. Parametri voi olla myös jokin lähtöarvo, kuten kerroskorkeus. Algoritmi on monitahoinen käsite, mutta tämän työn yhteydessä se tarkoittaa määrättyä tehtäväsarjaa, joka luodaan koodin avulla. Tehtäväketjun jokainen osa on yksiselitteisesti määritetty ja niillä on tietty tarkoitus algoritmissa.

Tutkimukseen vaikutti usea motivaatiolähde. Työntekijän oma kiinnostus tietomallintamiseen on lähtökohtana koko diplomityölle. Työntekijä teki kandidaatintyönsä tietomallien laadunvarmistuksesta ja on työskennellyt Ramboll Finland Oy:ssä kaksi vuotta elementtisuunnittelun parissa, jonka avulla tietomallintaminen ja elementtisuunnittelu on tullut hyvin tutuksi. Elementtisuunnittelua tehdessä on ajauduttu usein tilanteisiin, joissa on mietitty olisiko olemassa tehokkaampaa tapaa toteuttaa suunnittelua.

Rambollissa työskentelee sekä Suomessa että globaalisti useita algoritmiavusteisen suunnittelun asiantuntijoita. Tällä hetkellä Suomessa algoritmiavusteista suunnittelua hyödynnetään muun muassa arkkitehti-, rakenne- ja infrasuunnittelussa. Rambollissa isona teemana on digitalisaation jatkuva kehittäminen ja hyödyntäminen suunnittelutyössä. Diplomityön raamit alkoivat muodostua näistä kahdesta motivaationlähteestä. Hyvin nopeasti ensimmäisten palaverien jälkeen diplomityön aihealue ja painopiste muodostuivat. Työssä on tarkoituksena tutkia mahdollisuutta kehittää elementtisuunnittelua algoritmiavusteisen suunnittelun avulla.

1.2 Tutkimuksen tavoitteet, menetelmät ja rajaukset

Tutkimuksessa on tehty muutamia rajauksia aihealueeseen ja tutkittaviin ilmiöihin. Yhtenä tärkeänä rajauksena voidaan pitää elementtisuunnittelua. Toisena merkittävänä valintana on tutkia case-tutkimuksessa ainoastaan betonisia kantavia väliseinäelementtejä. Nämä ohjaavat työn pääpainoa elementtisuunnitteluun, eikä niinkään rakennesuunnittelua kohti. Rajaus tiettyyn rakennusosaan tehtiin, jotta algoritmiavusteiseen suunnitteluun liittyviä yksityiskohtaisia piirteitä kyetään tutkimaan riittävän tarkasti. Ohjausryhmän keskusteluiden perusteella todettiin, että rakennusosasta riippumatta tietyt erikoispiirteet algoritmiavusteisessa suunnittelussa toistuvat. Esimerkkinä toistuvasta piirteestä voidaan pitää lähtötietovaatimusten erikoispiirteitä verrattuna perinteiseen mallintavaan suunnitteluun. Näitä toistuvia piirteitä voidaan pitää välillisesti arvoa tuottavina. Vaikka lähtökohtaisesti case-tutkimuksessa tutkitaan tiettyä elementtityyppiä, kuitenkin välillisesti tutkimuksella tuotetaan yleistettävää tietoa algoritmiavusteiseen elementtisuunnitteluun.

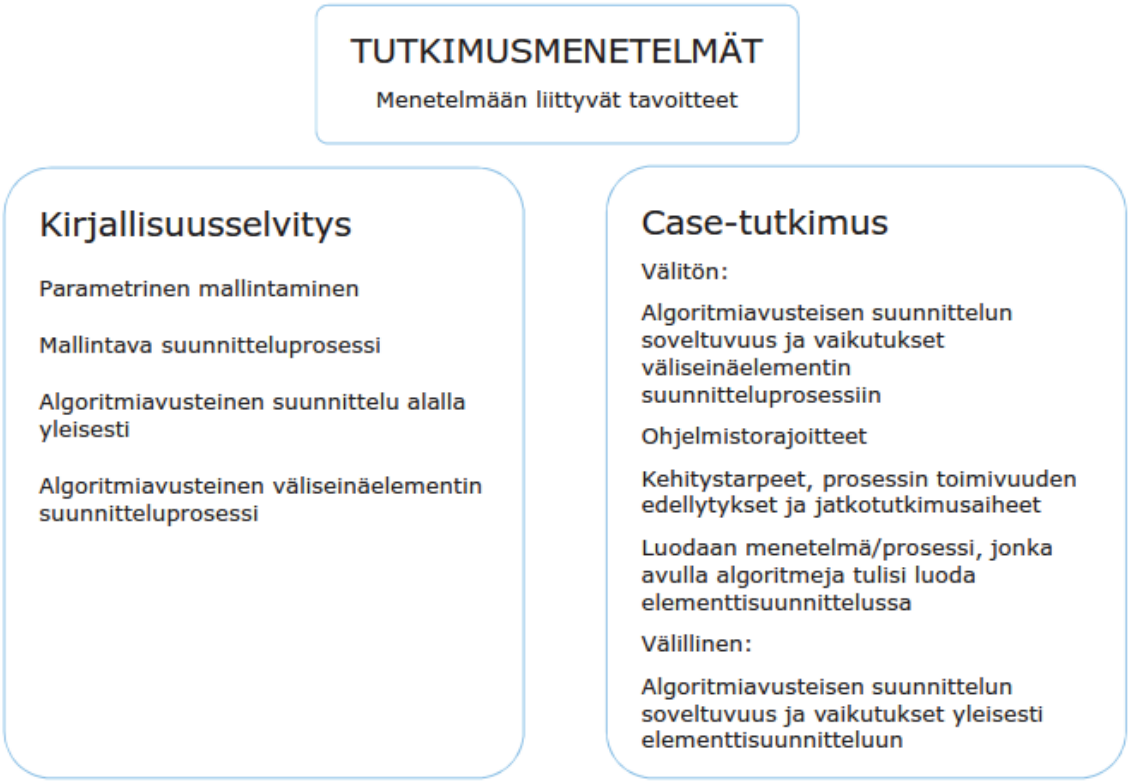
Toisena syynä rajaukselle oli, että väliseinät ovat työn ajankäytön kannalta järkeviä, sillä niiden suunnitteluprosessi tunnetaan todella hyvin, jolloin riski ajan riittämättömyydestä minimoitiin. Tämän ei kuitenkaan oleteta vaikuttavan työn laatuun, sillä väliseinien suunnittelu on elementtisuunnittelussa hyvin tyypillistä ja niiden suunnitteluun liittyy olennaisesti myös muut suunnittelualat. Koettiin, että näillä rajauksilla saadaan mahdollisimman kattavasti kerättyä tietoa siitä, kuinka algoritmiavusteinen suunnittelu sopii sekä väliseinien suunnitteluun että elementtisuunnitteluun yleisellä tasolla.

Kuten mainittu, tutkimuksella on kaksi rinnakkaista suuntaa: eksaktimpi tutkimus väliseinäelementin algoritmiavusteisesta suunnittelusta ja sekä välillisesti että välittömästi tutkia algoritmiavusteisen suunnittelun vaikutuksia elementtisuunnitteluun yleisellä tasolla. Tästä muodostuu yksi päätavoite: selvittää algoritmiavusteisen suunnittelun hyödyntämisen vaikutukset sekä betonisten väliseinäelementtien suunnitteluprosessiin että yleisesti elementtisuunnitteluun. Päätavoite jakaantuu useampaan pienempään tavoitteeseen:

- Tiedolliset tavoitteet
    - Betoniväliseinäelementin mallintavan suunnitteluprosessin kuvaaminen
    - Betoniväliseinäelementin algoritmiavusteisen suunnitteluprosessin kuvaaminen
    - Algoritmiavusteisen suunnittelun vaikutukset betonisten väliseinäelementtien suunnitteluprosessiin
        - Hyödyt, haitat, ongelmakohdat ja kehitystarpeet
    - Ohjelmistoihin liittyvät rajoitteet
        - Rajapinnat ja ohjelmistojen sisäiset ongelmakohdat
    - Algoritmiavusteisen suunnittelun hyödyntäminen yleisesti rakennusalalla
    - Jatkotutkimusaiheet

- Teollisia tavoitteita
    - Betoniväliseinäelementin algoritmiavusteisen suunnitteluprosessikaavion luominen
    - Kehittää uudelleen hyödynnettävä menetelmä algoritmin luomiseen elementtisuunnittelussa
        - Case-tutkimuksessa luotavan algoritmin avulla pyritään löytämään tietyt selkeät osa-alueet, joiden avulla voidaan luoda elementtisuunnitteluun yleistettävä prosessikaavio

Tietomallinnusohjelmiston rajaus Tekla Structuresiin (Tekla) oli selkeä ratkaisu, sillä Rambollissa Tekla on pääsääntöinen työkalu elementtisuunnittelussa. Ohjelmistoriippumaton tutkimus haluttiin pitää osana tutkimusta, koska ei haluttu ainoastaan keskittyä jonkin tietyn ohjelmiston mahdollisuuksiin tai rajoitteisiin. Koska Tekla valittiin tietomallinnusohjelmistoksi, valittiin algoritmien luomiseen Rhinoceros 3D (Rhino) ja Grasshopper -ohjelmistojen yhdistelmä. On kuitenkin hyvä huomioida, että tutkimus olisi voitu yhtä hyvin toteuttaa käyttämällä Autodeskin tietomallinnusohjelmistoa Revittiä ja algoritmien luomiseen Revitin lisäosaa Dynamoa.



Kuva 1. *Tutkimusmenetelmien ja tutkimuksen tavoitteiden välinen yhteys.*

Tutkimuksessa käytetään kahta tutkimusmenetelmää kuvan 1 mukaisesti. Kirjallisuusselvitys on ensimmäinen tutkimusmenetelmä. Kirjallisuusselvityksen avulla tehdään kattava taustaselvitys, jonka avulla saadaan tarvittava teoriatieto toteuttaa aiheeseen liittyvä case-tutkimus. Sen avulla tutkitaan parametrista mallintamista, mallintavaa

suunnitteluprosessia, algoritmiavusteista suunnittelua alalla yleisesti sekä algoritmiavusteista väliseinäelementin suunnitteluprosessia. Toisin sanoen, kirjallisuusselvityksellä on tarkoitus tuottaa välitöntä tietoa sekä perinteisestä että algoritmiavusteisesta väliseinäelementtien suunnitteluprosessista.

Case-tutkimuksen tarkoituksena on testata luotua suunnitteluprosessia. Välittömästi saadaan tietoa algoritmiavusteisen suunnittelun soveltuvuudesta ja vaikutuksista väliseinäelementtien suunnitteluprosessiin, algoritmiavusteiseen suunnitteluun liittyvien ohjelmistojen rajoitteita, prosessin toimivuuden edellytyksiä ja luodaan prosessi, jonka avulla algoritmeja tulisi luoda elementtisuunnittelussa. Välillisesti saadaan tietoa myös algoritmiavusteisen suunnittelun soveltuvuudesta yleisesti elementtisuunnitteluun.

1.3 Tutkimuksen toteutus ja rakenne

Tutkimuksen rakenne on toteutettu Tampereen teknillisen yliopiston opinnäytetyöohjeen mukaisesti: tutkimuksen tausta, teoria tutkimuksen taustalla, tutkimusmenetelmät, tulosten analysointi ja johtopäätökset. Ensimmäisessä luvussa esitetään tutkimuksen tausta, motivaatio, tavoitteet, menetelmät, rajaukset ja rakenne. Toinen, kolmas ja neljäs luku on osa kirjallisuusselvitystä. Toisessa osiossa perehdytään parametrisen suunnittelun taustoihin, käsitteisiin ja käyttömahdollisuuksiin. Kolmannessa osassa tutkitaan tarkemmin algoritmeja, niiden luomista, algoritmiavusteista suunnittelua, ohjelmistoja ja niiden sovellutuksia. Neljännessä osassa perehdytään elementtisuunnitteluun, tutkitaan elementtisuunnittelun taustoja ja nykyistä elementtisuunnittelun prosessia. Lisäksi luodaan algoritmiavusteinen väliseinäelementtien suunnitteluprosessi.

Viides luku on case-tutkimuksen toteutus. Siinä testataan ja tutkitaan edellisessä osiossa luotua algoritmiavusteisen väliseinäelementin suunnitteluprosessia. Kuudennessa osiossa analysoidaan case-tutkimuksen tuloksia. Lopuksi viimeisessä osiossa tehdään johtopäätökset tutkimuksesta. Arvioidaan kriittisesti tutkimuksen tavoitteiden saavuttamista, pohditaan mahdollisia jatkotutkimusaiheita ja tehdään yhteenveto tutkimuksesta.

2. PARAMETRINEN SUUNNITTELU

Tietomallintamisen on ajateltu aikaisemmin olevan suunnittelusta erillinen asia. Tästä on seurannut, että on haluttu siirtyä käyttämään termiä mallintava suunnittelu. Se kuvaa paremmin tämän hetken tilannetta, jossa tietomallit toimivat suunnittelijoiden työkaluina. Näiden työkalujen avulla on löydetty uusia tapoja työskennellä ja suunnitella.

Tutkimuksen taustana ja motivaationa on sekä tutkijan että tilaajayrityksen Ramboll Finland Oy:n mielenkiinto tutkia parametrisen mallintamisen ja algoritmiavusteisen suunnittelun yhteiskäytön mahdollisuuksia. Tässä osiossa tutkitaan parametrisen mallintamisen käsitteitä, käytettyjä ohjelmistoja ja merkittäviä julkaisuja ja vaatimuksia alalla.

2.1 Tietomallinnus

Työssä tutkitaan tietomallintamista ja Tekla Structures -ohjelmiston käyttöä elementtisuunnittelussa. Tekla on rakenteiden tietomallintamiseen kehitetty ohjelmisto (Eastman et al. 2008, p. 29, 40, 61). Teklalla tehdään mallintavaa suunnittelua, jonka tarkoituksena on tuottaa rakennuksen toteuttamista varten suunnitelmia, kuten elementtikuvia ja kaavioita. Tekla mahdollistaa sekä paikallavalurakenteiden että elementtirakenteiden valmistuspiirustusten luomisen. Sitä käytetään myös teräsrakenteiden suunnittelussa. Ohjelmistolla kyetään mallintamaan raudoitteet, liitoskappaleet, valutarvikkeet, muotit, ynnä muut. Kappaleille pystytään asettamaan erilaisia suunnittelua, rakentamista ja toteutusta helpottavia tietoja (Eastman et al. 2008, p. 29-32, 40-43).

Nykymallinen Tekla pohjautuu Teklan X-steel -ohjelmistoon (Eastman et al. 2008, p. 40). Teklan omistaa nykyään Trimble Solutions. Ohjelmisto on elementtirakentamisen tarpeisiin soveltuva, mutta palvelee myös paikallavalurakenteiden ja teräsrakenteiden suunnittelua. Teklaa käytetään myös talotekniikka- ja infra-alan suunnittelun työkaluna (Mallintava suunnittelu; Eastman et al. 2008, p. 29,40,61).

Tietomallintamista edeltävät suunnittelutyökalut ovat Computer-Aided Design (CAD) perustuvia ohjelmistoja. Kehittyneempiä CAD ohjelmistoja on edelleen käytössä tukemassa suunnittelutyötä. CAD ohjelmistoilla kyetään tuottamaan suunnitelmia, jotka muodostuvat pääosin viivoista, viivatyypeistä ja niihin liitetyistä tiedoista (Eastman et al. 2008, s. 12). Hiljalleen muodostui tarve lisätä CAD tiedostoihin rakennusosaan liittyvää tietoa ja esimerkiksi pintoja. Tämä johti ajattelumalliin, jossa keskityttiin tietoon, eikä 2D tai 3D kuviin. Näistä tarpeista kehittyi idea tietomalleista (Kensek & Noble 2014, p. xxiii-xxix; Tanska & Österlund 2014, p. 17).

Ensimmäiset tietomallintamiseen pohjautuvat ohjelmistot kehittyivät CAD suunnittelun puutteiden pohjalta vuosikymmeniä sitten (Eastman et al. 2008, p. 12-13, 26-27). Huolimatta siitä, että tietomallintaminen on mullistanut rakennusalaa jo useita vuosikymmeniä, ei alalla ole täysin vakiintunutta määritelmää tietomallille. M.A. Mortenson Companyn mukaan tietomallin tieto sisältää kuusi vaatimusta:

- Digitaalinen
- Avaruudellinen (3D)
- Mitattavissa oleva
- Ymmärrettävä
- Saatavilla oleva
- Aikaa kestävä

Kuten jo Rakennusteollisuuden (2005) tutkimuksessa todettiin, tietomallia voidaan kuvata rakennuksen tuotetietomallina. Tämä on varsin kuvaava nimi, sillä se kertoo paremmin siitä, että tietomallin tarkoitus on sisältää kaikkiin siinä oleviin tuotteisiin tai rakennusosiin liittyvää tietoa. Tietomallin tulee kuvata rakennuksen kaikkien osien tiedot jäsennellysti. Todetaan kuitenkin, että tuotetietomalli tai tuotemalli ei ole erityisen vakiintunut käsite. Vakiintuneempana käsitteenä voidaan pitää tietomallia (Mallintava suunnittelu).

2.2 Tietomallintamiseen liittyvät tärkeimmät julkaisut ja ohjeistukset

Rakennusteollisuuden julkaisussa (Pro IT -tutkimus) muun muassa avataan tietomallinnukseen liittyviä käsitteitä. Building Information Model (BIM) eli tietomalli tunnetaan myös rakennuksen tuotetietomallina (Mallintava suunnittelu). Tämä on varsin kuvaava nimi, sillä se kertoo paremmin siitä, että tietomallin tarkoitus on sisältää kaikkiin siinä oleviin tuotteisiin tai rakennusosiin liittyvää tietoa. Tietomallin tulee kuvata rakennuksen kaikkien osien tiedot jäsennellysti. Todetaan kuitenkin, että tuotetietomalli tai tuotemalli ei ole erityisen vakiintunut käsite. Vakiintuneempana käsitteenä voidaan pitää tietomallia.

Pro IT -tutkimuksen ideana oli kehittää alalle yhteinen tapa tuottaa tuotetietomalleja, jotka palvelisivat mahdollisimman hyvin projektien eri osapuolia. Tutkimuksessa oli kolme tavoitetta: luoda kaikille yhteinen mallinnuskäytäntö, tiedonsiirron sujuvoittaminen, tuotemallien nimikkeistöjen ja esitystapojen yhtenäistäminen. Tutkimuksen tuotoksena tuli useampia ohjeita, prosessikuvauksia ja selvityksiä. Tutkimuksesta toteutettiin myös pilottihankkeita, joissa testattiin tutkimuksessa tuotettuja ohjeistuksia ja uusia suunnitteluprosesseja. Pilottihankkeista kaksi oli asuntokohteita, yksi myymälärakennuskohde ja yksi paikoitushalli. Näistä pilottihankkeista selkeimpinä hyötyinä nähtiin olevan kolmiulotteisten suunnitelmien havainnollisuus, tuotetietomallin hyödynnettävyys arkkitehtisuunnittelussa, rakennesuunnittelussa ja elementtisuunnittelussa. Näiden

lisäksi eri suunnittelualojen suunnitelmien yhteensovittaminen helpottui huomattavasti, määrä- ja kustannuslaskennan tarkkuus parani ja toteuttajalle saatiin huomattavasti virheettömämpiä piirustuksia ja listauksia (Pro IT -tutkimus; Mallintava suunnittelu).

Siitä mihin Pro-IT -hanke loppui, alkoi alalla iso kehitys tietomallintamisessa. Senaatti-kiinteistöt julkaisi 2007 vuonna tietomallivaatimukset, jonka pohjalta vuonna 2011 aloitettiin COBIM-hanke. Hankkeessa toimi rakennusteollisuuden alalta usea tekijä, kuten Senaatti-kiinteistöt, buildingSMART Finland, Espoon ja Helsingin kaupunkien Tilakeskukset, sekä isoja rakentajia ja suunnitteluyrityksiä. Hanketta johti Rakennustietosäätiö RTS. Lähtökohtana hankkeelle oli rakennusalalla tapahtuva nopeasti kasvava tietomallintamisen lisääntyminen. Alan toimijat halusivat luoda Pro-IT -hanketta tarkemmat yhteiset pelisäännöt, vaatimukset ja ohjeet tietomallinnettujen hankkeiden toteuttamiseen. Pro-IT -hankkeessa ohjeet eivät olleet riittävän tarkkoja rakennushankkeiden eri osapuolten tukemiseen nopeasti muuttuvalla toimialalla. Lopputuloksena COBIM-hankkeelle oli Yleiset Tietomallivaatimukset 2012 -julkaisusarja. Julkaisusarja muodostuu alkuperäisistä 1-9 osista ja myöhempinä osina julkaistuista 10-14 osista (Yleiset Tietomallivaatimukset 2012).

Osa 1 yleinen osuus -teos sisältää tietomallintamiseen liittyviä yleisiä tavoitteita ja vaatimuksia. Siinä käydään läpi myös eri hankevaiheisiin liittyvät tietomallit ja niihin liittyvät vaatimukset. Näiden lisäksi kerrotaan malliteknisistä asioista, kuten koordinaatistosta, mittatarkkuudesta, nimeämisestä, ja vastaavista. Tietomalleille asetettavia yleisiä tavoitteita:

- Tukea hankkeen päätöksentekoprosesseja
- Sitouttaa osapuolet hankkeen tavoitteisiin mallin avulla
- Havainnollistaa suunnitteluratkaisuja
- Auttaa suunnittelua ja suunnitelmien yhteensovittamista
- Nostaa ja varmistaa rakennusprosessin ja lopputuotteen laatua
- Tehostaa rakentamisaikaisia prosesseja
- Parantaa turvallisuutta rakentamisen aikana ja elinkaarella
- Tukea hankkeen kustannus ja elinkaarianalyysejä
- Tukea hankkeen tietojen siirtämistä käytönaikaiseen tiedonhallintaan

Vaatimukset kattavat sekä uudis- että korjausrakentamiskohteet koko niiden elinkaaren aikana. Ensimmäiset yhdeksän osaa antavat kuitenkin tarkemmat vaatimukset hankkeen eri osapuolille, tiedonhallinnalle, laadunvarmistukselle ja vastaaville. Yksityiskohtaiset vaatimukset tulee kuitenkin sopia aina projektikohtaisia (Yleiset Tietomallivaatimukset 2012).

Osa 2 lähtötilanteen mallinnus -teoksessa annetaan vaatimuksia inventointimallin luomiseen. Siinä käsitellään yksityiskohtaisesti lähtötilanteen mallintamiseen kuuluvia erilaisia mittauksia, tutkimuksia ja tuotettavia dokumentteja. Näihin annetaan tietosisältöjä tarkkuustasovaatimuksia. Tarkkuustasoille annetaan kolme eri tasoa:

1. Tilamalli
    a. Tilat ja rakenteet mallinnetaan karkealla tasolla
2. Rakennusosamalli
    a. Tarkkuustaso riittävä ehdotussuunnitelmatasoisiin hankesuunnitelmiin
3. Rakennusosamalli
    a. Detaljien tarkkuutta ja mallinnettavien kappaleiden määrää on kasvatettu taso kahden tarkkuuteen nähden

Teoksessa annetaan myös mallinnusvaatimukset hankkeen eri vaiheille (Yleiset Tietomallivaatimukset 2012).

Kun ensimmäinen osa antoi yleisiä tietomallintamiseen liittyviä vaatimuksia, antavat osat 3-5 tarkempia vaatimuksia suunnitteluosapuolille. Osa 3 arkkitehtisuunnittelu -teos antaa arkkitehdeille valmiuksia tietomallinnettujen hankkeiden hallintaan. Arkkitehtisuunnitteluun liittyvien mallinnusperiaatteiden lisäksi teoksessa annetaan hankkeen eri vaiheisiin liittyviä tietomallinnusvaatimuksia. Arkkitehtimalleille on annettu kolme eri tarkkuustasoa. Teoksen lopussa on Talo 2000 -nimikkeistön mukainen luettelo, jossa on myös hankevaiheet esiteltynä. Luetteloon on merkattu jokaisen rakennusosan kohdalle pakollinen tai suositeltava tarkkuustaso jokaiselle hankkeen eri vaiheelle (Yleiset Tietomallivaatimukset 2012).

Osa 4 talotekninen suunnittelu -teos antaa samalla tavalla kuin arkkitehtisuunnitteluteoksessa yksityiskohtaisia vaatimuksia kyseisen suunnittelualan tietomalliteknisiin seikkoihin. Teos sisältää erikseen vaatimuksia liittyen tilavarauksiin, LVI-suunnitteluun, rakennusautomaatioon, sähkö- ja telesuunniteluun. Oleellisena osana talotekniseen suunnitteluun kuuluu tiedon yhteensovittaminen ja siihen liittyvät vaatimukset, joihin teos antaa ohjeita ja hyviä toteutustapoja. Samalla tavalla kuin arkkitehtisuunnittelu teoksessa, myös tämän teoksen loppuosassa on tarkka listaus hankkeen eri vaiheiden mukaisesti esitetyistä talotekniikan tarvikkeiden mallinnustarkkuuksista (Yleiset Tietomallivaatimukset 2012).

Osa 5 rakennesuunnittelu -teos antaa työkaluja mahdollistamaan rakenne- ja elementtisuunnittelussa toteutettujen tietomallien hyödyntämisen mallinnetuissa projekteissa. Siinä annetaan ohjeita ja määräyksiä yhteisistä mallinnuskäytännöistä. Tässäkin osassa esitetään hankkeen eri vaiheisiin liittyviä mallinnusteknisiä vaatimuksia. Teoksen liitteenä on yksityiskohtaiset listaukset suunnitteluhankkeiden eri vaiheiden tietomallien sisällöstä ja sen tarkkuustasosta. Lisäksi on myös esimerkit tietomalliselostuksesta ja tietomallin tarkastuslomakkeesta (Yleiset Tietomallivaatimukset 2012).

Osa 6 laadunvarmistus -teos käsittelee laadunvarmistusta ja -tarkastusta hankkeen eri osapuolien näkökulmista. Pääasiallisena tarkoituksena teoksella on esittää tietomallinnettuissa hankkeissa mallin tietosisällön vaatimuksia ja antaa ohjeita tietosisällön tarkastamiseen. Nämä hankkeen alussa määrätyt vaatimukset ja tarkastukset koskevat sekä hankkeen kokonaisuuden hallintaa että hankkeen jokaista osapuolta. Yhtä lailla ne kos-

kevat myös jokaista mallissa työskentelevää tai mallia käyttävää yksittäistä suunnittelijaa. Teos antaa siis määräyksiä, mutta niiden lisäksi paljon ohjeita ja työkaluja määrättyjen tavoitteiden saavuttamiseen. Näitä työkaluja ovat esimerkiksi erilaiset tarkastus- ja tehtävälistat ja lomakkeita (Yleiset Tietomallivaatimukset 2012).

Osa 7 määrälaskenta -teos antaa muiden teosten tapaisesti kyseiseen asiaan liittyviä malliteknisiä vaatimuksia. Teoksessa esitetään määrälaskentaan liittyviä käsitteitä ja menetelmiä. Näiden lisäksi teoksessa kuvataan määrälaskennan prosessi ja tyypillisiä määrälaskentaan liittyviä ongelmakohtia. Liitteenä on myös Talo 2000 hankenimistön mukainen luettelo rakennusosista ja kenen ensisijaisella ja toissijaisella vastuulla on esittää ne mallissaan (Yleiset Tietomallivaatimukset 2012).

Osa 9 Mallien käyttö talotekniikan analyyseissä -teos antaa kattavan kuvauksen erilaisten analyysien vaihtoehdoista ottamatta kantaa niihin käytettävistä ohjelmistoista. Teoksessa esitetään analyyseiksi muun muassa energia- ja olosuhdesimuloinnit, virtaussimulointi, talotekniikan elinkaarikustannusten analyysi, ympäristövaikutusanalyysi, tekniset havainnollistamiskuvat, valaistuslaskenta ja -visualisointi, valaistussimulointi ja TATE-järjestelmäanalyysit. Siinä kerrotaan myös analyysien suorittamiseen liittyvistä yksityiskohdista, kuten lähtötiedoista ja niiden laadusta, ja tulosten havainnollistamisesta (Yleiset Tietomallivaatimukset 2012).

Osa 10 Energia-analyysit -teoksessa tarkastellaan tarkemmin yhtä analyysivaihtoehtoa, energia-analyysia. Siinä käsitellään sekä suunnittelun että rakentamisen kannalta energiatehokkuutta ja sisäolosuhteiden hallintaa. Teos antaa vaatimukset tietomallien hyödyntämiseen energia-analyysejä varten hankkeen eri vaiheissa. Teoksessa esitellään myös alalla yleisesti käytössä olevia energia-analyysiohjelmia, ja niihin liittyviä tiedonsiirtovaatimuksia (Yleiset Tietomallivaatimukset 2012).

Osat 8, 11, 12, 13 ja 14 jätetään tarkastelematta, koska ne eivät liity diplomityön sisältöön. YTV 2012 on hyvä lähtökohta tietomallinnettujen hankkeiden hallintaan. Tärkeintä on projektin kannalta kaikkien osapuolten mallintamiskäytännöistä ja -vaatimuksista yhdessä sopiminen ja toisten osapuolten vaatimusten ymmärtäminen. Tämä koskee sekä sopimusteknisiä asioita, että yhteisiä pelisääntöjä ja kaikkien osapuolten asettamia tavoitteita. On tärkeää ymmärtää miksi tiettyjä vaatimuksia annetaan, ja mihin se vaikuttaa, jos joku tai jotkut osapuolet eivät jostakin syystä ota huomioon sovittuja vaatimuksia. Nämä asiat korostuvat varsinkin mallintavassa suunnittelussa, kun suunnittelua tehdään eri ohjelmistoilla ja tieto tulee saada siirtymään ohjelmistosta toiseen (Harmanen 2010).

2.3 Objektien parametrinen mallintaminen

Teoksessa (Eastman et al. 2008) tietomallintaminen esitetään parametrisena kappaleiden mallintamisena. Parametrisuus ajatellaan muodostuvan kappaleisiin syötettävistä tie-

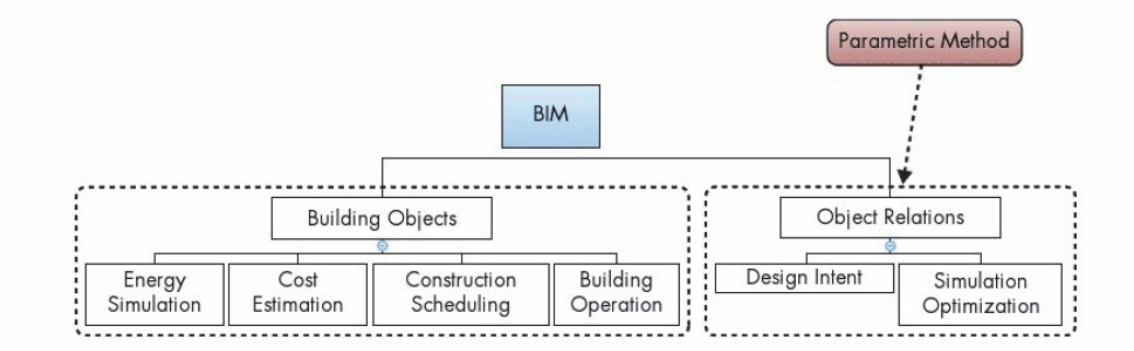
doista ja yhteyksistä. Tiedot syötetään kappaleille niin, että ne muodostavat yhteyksiä ja sääntöjä – eräänlaisia hierarkioita. Sääntöjä ja yhteyksiä voi olla usean kaltaisia. Ne voivat olla kappaleiden välisiä tai kappaleen sisäisiä. Tällaisia ovat esimerkiksi etäisyyksiin, kulmiin tai pintoihin liittyviä parametrit.

Lyhykäisyydessään voidaan siis todeta, että tietomallit sisältävät hiearkisia tietorakenteita, joissa kappaleet sisältävät älykästä tietoa itsestään ja kappaleiden välisistä riippuvuussuhteista. Toisin sanoen nämä kappaleet ovat parametrisia. Parametriset kappaleet sisältävät parametrisia ominaisuuksia, geometriatietoja, kappaleeseen liittyvää tietoa ja sääntöjä ja tiedon tasoja. Säännöillä ja yhteyksillä mahdollistetaan toisiinsa liitettyjen kappaleiden automaattinen muokkaaminen. Parametrien tulee olla määritelty sellaisella tarkkuudella, että ne tunnistavat ja ilmoittavat käyttäjälle, kun tapahtuu sääntöä rikkova muutos. Kappaleeseen liitettyä tietoa on myös voitava hakea ja irrottaa siitä (Eastman et al. 2008, p. 13-15).

On tärkeää erottaa käsitteet parametrinen arvo (parametric variable) ja parametrinen ominaisuus (parametric property) toisistaan. Parametrinen arvo on muuttuja tai arvo, joka ei sisällä riippuvuussuhteita. Parametrinen ominaisuus kertoo arvojen tai kappaleiden välisistä riippuvuussuhteista (Trimble Solutions Corporation 2017). Parametrisella arvolla kuvataan ja voidaan muuttaa parametrista ominaisuutta. Teklan parametrisuuden periaatteena on luoda eri tasoisia riippuvuussuhteita kappaleille. Nämä tasot tunnetaan hierarkiatasoina. Kappaleille on asetettu tiettyjä ominaisia parametreja ja riippuvuuksia, ja käyttäjät voivat muuttaa näiden parametrien arvoja. Käyttäjät voivat lisätä myös parametrisia ominaisuuksia esimerkiksi komponenttien avulla. Nämä komponentit voivat määrittää kappaleiden välisiä suhteita tai kappaleen sisäisiä suhteita. Parhaimmillaan riippuvuussuhteet keskustelevat eri komponenttien kesken ja muodostavat lopputuloksen näiden sääntöjen perusteella ottaen huomioon kyseisten sääntöjen hierarkiatasot (Eastman et al. 2008; Trimble Solutions Corporation 2017).

Teoksessa (Eastman et al. 2008, p. 29-32) esitetään Teklan parametrisuudesta hyvä esimerkki. Siinä selitetään seinäobjektin sisältöä ja sisällön muodostamisen logiikkaa. Monimuotoinen seinäobjekti sisältää useita pintoja, joista osa voidaan sitoa parametrisilla ominaisuuksilla haluttuihin kappaleisiin. Samaan aikaan osa pinnoista voi olla parametrisella arvolla kiinteästi määrätty. Esimerkiksi parametrinen ominaisuus seinän korkeus voidaan säännöllä sitoa alkamaan halutun kerroksen lattiatasosta ja päättyä ylemmän kerroksen lattian alapintaan. Toisaalta korkeus voidaan määrittää halutulla parametrisella arvolla, jolloin sitä ei ole sidottu parametrisesti mihinkään toiseen kappaleeseen. Ensimmäisessä tapauksessa, jossa seinän korkeus on sidottu ylempiin ja alempiin rakenteisiin parametrisella ominaisuudella, kerroskorkeuden muuttuessa seinän korkeus muuttuu automaattisesti mukana. Jälkimmäisessä tapauksessa seinä ei seuraa ylempiä ja alempia rakenteita, vaan sen arvo on käsin muutettava. Seinä voi sisältää myös kappaleita, kuten ikkunoita, jotka on sidottu kyseisen seinän sisäiseen koordinaattijärjestelmään.

Seinä sisältää myös muita parametrisia ominaisuuksia, kuten seinän materiaali, jolle voidaan antaa parametrinen arvo. Käytännössä kaikki tieto, jotka annetaan seinäkappaleelle ovat parametreihin liittyviä parametrisia arvoja. Seinän nimeäminen, class, numbering prefix ja muut vastaavat ovat parametreja, joille on annettava arvo. Loppujen lopuksi ajatellaan seinän olevan tietokanta, joka sisältää tietoa, jotka ovat useissa tasoissa ja sisältävät monen tasoisia sääntöjä. Teoksessa todetaan vielä, että ilman tietomallien parametrisuutta, jotka mahdollistavat sekä pienien että isojen asioiden automaattisen päivittämisen sekä hierarkkisen tietokannan luomisen, ei tietomallintaminen olisi todennäköisesti kannattavaa (Eastman et al. 2008, p. 29-32).



Kuva 2. *Parametrisuuden yhteys tietomallintamiseen (Kensek & Noble 2014, p. 61).*

Kuvassa 2 on havainnollistettu tietomallin sisältämien kappaleiden tiedon ja parametrisuuden suhdetta. Kuvassa parametrisuus esitetään erillisenä osana kappaleita. Kuvan sisällön voisi paremminkin esittää niin, että tietomallissa olevat kappaleet sisältävät tietoa sen geometriasta, materiaaleista ja vastaavista, johon on lisätty kappaleiden välisiä parametrisia ominaisuuksia (Kensek & Noble 2014, p. 59-61).

3. ALGORITMIAVUSTEINEN MALLINNUS

”Parametrinen mallinnus tarkoittaa riippuvuussuhteiden rakentamista suunnittelumallin eri geometristen osien ja algoritmia ohjaavien parametrien välille.” (Tanska & Österlund 2014, p. 13), sitaatti kiteyttää lyhyesti parametrisen mallintamisen. Parametrinen mallintaminen on osa algoritmiavusteista mallintamista. Algoritmiavusteisen mallintamisen yhtenä tavoitteena on siirtyä yhden ratkaisun ajattelutavasta tapaan, jossa pyritään luomaan systeemi tai alusta, jonka avulla voidaan luoda useampi suunnitteluratkaisu (Kensek & Noble 2014, p. 60). Tulevassa osiossa tutkitaan sekä algoritmiavusteista mallintamista että suunnittelua. Lisäksi perehdytään lyhyesti sekä tutkimuksessa että alalla yleisesti käytettäviin ohjelmistoihin.

3.1 Periaate ja tausta

Edellisessä osiossa tutkittiin parametrisuuden käsitettä. Lopputuloksena sen voidaan ajatella olevan pienten asioiden välisten riippuvuussuhteiden luomista tai kokonaisten kappaleiden välisten riippuvuussuhteiden luomista (Eastman et al. 2008, p. 29-35). Edellisessä osiossa todettiin myös, että Tekla on parametrista periaatetta hyödyntävä ohjelmisto. Tässä diplomityössä tutkitaan parametrista mallintamista Teklassa hyödyntämällä Grasshopperin visuaalista koodausta. Puhutaan myös visuaalisesta skriptauksesta (Tanska & Österlund 2014, p. 13). Vaikka pääsääntöisesti keskitytään tiettyihin ohjelmistoihin, tarkoituksena on myös tutkia algoritmiavusteista suunnittelua ja mallintamista ohjelmistoriippumattomista näkökulmista.

Visuaalinen skriptaus on yksinkertaisuudessaan koodaamista, jossa ideana on käyttää valmiita visuaalisia ohjelmakomponentteja, joita linkitettäessä toisiinsa luodaan algoritmi. Algoritmi käsitteenä on monitulkintainen ja se on myös hyvin riippuvainen kontekstista. Käsitteellä on matemaattinen tausta, mutta nykyään se paremminkin käsitetään ohjelmointiin liittyväksi käsitteeksi. Algoritmi voidaan ymmärtää reseptinä tai tehtäväsarjana, jonka tarkoituksena on tuottaa jokin ennalta määrätty lopputulos. Se on ketju käskyjä ja arvoja, jotka muodostavat kokonaisuuden. Tässä työssä sen voidaan ajatella olevan ketju, jossa tieto virtaa sen lävitse. Eli algoritmi muodostuu käskyistä, tehtävistä ja arvoista, joissa jokaisella osalla on oma tehtävänsä ketjussa. Jokaisen ketjun osan tulee olla tarkasti määrättyjä. Vaikka algoritmit ovat keinoja saada yksi tehtäväsarja loppuun, ei ole yhdentekevää millä tavalla tämä algoritmien kokonaisuus muodostetaan (Woodbury 2010, p. 11-16, 34-35; Tanska & Österlund 2014, p. 20).

Koska algoritmiavusteinen suunnittelu tai parametrinen mallintaminen ovat käsitteinä ja asioina melko uusia Suomessa, ei alalle ole muodostunut selkeitä käsitteitä kuvaamaan

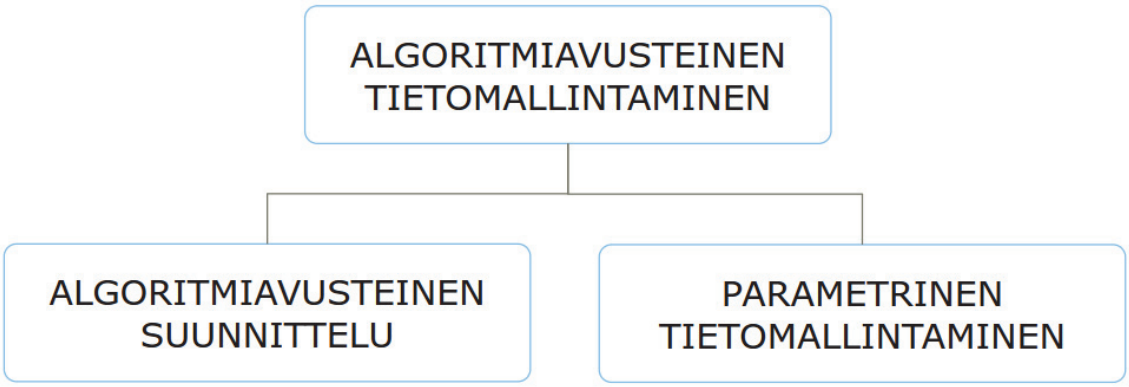
niitä. Käsitteet ovat olemassa, mutta niitä käytetään hyvinkin ristiriitaisesti. Tanska ja Österlund (2014) puhuvat sekä parametrisesta mallintamisesta että algoritmiavusteisesta suunnittelusta. Heidän työssään algoritmiavusteinen suunnittelu ymmärretään enemmänkin ajatustavan muutoksena, eli uutena tapana käyttää erilaisia algoritmeja hyödyntäviä menetelmiä perinteisen suunnittelun kehittämiseksi. Parametrinen mallintaminen voi olla yksi osa algoritmiavusteista suunnittelua.

Algoritmiavusteisesti luotu parametrinen malli on täysin dynaaminen, eli algoritmin avulla uudelleen ohjattavissa. Muutokset algoritmiin tai sen sisältämiin parametreihin eli muuttujiin muuttavat automaattisesti tietomallia. Kuitenkin aikaisemmin on todettu (Eastman et al. 2008; Woodbury 2010; Davis 2013) että parametrinen mallintaminen käsitteenä voidaan ajatella olevan pelkkiä riippuvuussuhteita ja arvoja ilman mitään yhteyttä algoritmiavusteisiin menetelmiin. Termit ja niiden käyttö ovat ristiriitaisia, varsinkin kun niiden käyttö englannin kielessä mielletään erilaiseksi mitä suomennetussa versiossa.

Humppi (2015) käyttää käsitteitä:

- Parametrinen mallintaminen (tietomallintaminen)
- Algoritmiavusteinen suunnittelu
- Algoritmiavusteinen mallintaminen (tietomallintaminen)

Työssään hän huomaa edellä mainitun käsitteiden ristiriitaisen tulkinnan. Hän kuvailee myös algoritmiavusteista suunnittelua. Parametrinen mallintaminen ja algoritmiavusteinen suunnittelu käsitteinä esitetään edellisissä teoksissa yhtenä ja samana asiana, vaikka ne eivät sitä ole. Humppi ehdottaakin uutta kuvaavampaa termiä, algoritmiavusteinen tietomallintaminen (Algorithm-Aided Building Information Modeling eli AAB). Termi on hyvin kuvaava, sillä ensinnäkin se kuvastaa tapaa tai työkalua jolla suunnittelutyötä tehdään ja toiseksi se kuvastaa mitä suunnittelutyöllä saavutetaan. Toisin sanoen se esittää menetelmän ja lopputuloksen.



Kuva 3. *Käsitekartta algoritmiavusteisesta mallintamisesta.*

Kuva 3 on käsitekartta, jonka tarkoituksena on selittää algoritmiavusteiseen mallintamiseen liittyviä käsitteitä ja selventää käsitteiden suhteet toisiinsa. Algoritmiavusteinen mallintaminen koostuu algoritmiavusteisesta suunnittelusta ja parametrisesta mallintamisesta. Molempia osa-alueita voidaan käyttää myös täysin erillään toisista. Kun parametrista mallintamista käytettään yksinään, puhutaan perinteisestä tietomallintamisesta. Vastaavasti algoritmiavusteinen suunnittelu voi olla mitä tahansa koodia hyödyntävää suunnittelua. Algoritmiavusteisessa mallintamisessa hyödynnetään molempia osa-alueita.

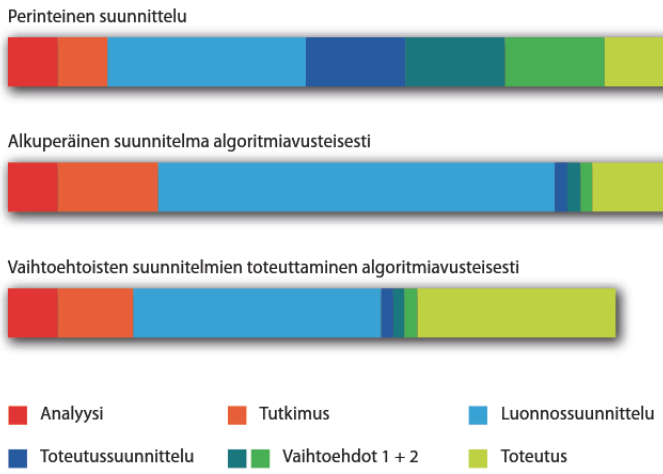
3.2 Algoritmiavusteinen suunnittelu

Algoritmiavusteinen mallintaminen tulee yhdistää algoritmiavusteisen suunnittelun ajattelutapaan. Analogia on sama kuin perinteisen mallintamisen ja mallintavan suunnittelun kanssa. Vaikka teoksissa puhutaan parametrisesta suunnittelusta, tässä työssä käytetään kuvaavampaa termiä algoritmiavusteinen suunnittelu. Nämä termit ja niiden ero on selitetty edellisessä osiossa.

Sekä algoritmiavusteinen mallintaminen että perinteinen mallintaminen ovat suunnittelijan työkaluja. Algoritmiavusteisessa suunnittelussa luodaan visuaalista skriptausta hyödyntävällä ohjelmistolla haluttu algoritmi, jonka avulla luodaan parametrinen malli. Huomattakoon, että tässä työssä tutkitaan nimenomaan visuaalista skriptausta osana suunnittelua, mutta algoritmiavusteinen suunnittelu voidaan toteuttaa myös täysin ilman visuaalista skriptausta. Osa käsiteltävistä asioista on myös yleistettävissä algoritmiavusteiseen suunnitteluun yleisemmällä tasolla. Tällä suunnittelumenetelmällä kyetään avaamaan lähes loputtomat suunnittelumahdollisuudet ja mahdollistetaan uudenlaisien suunnitteluprosessien hallitseminen (Tanska & Österlund 2014, p. 17-18).

Kirjallisuudessa esitetään algoritmiavusteisen suunnittelun nopeuttavan uusien suunnitelmien luomista ja tehokasta vaihtoehtojen vertailua. Perinteinen mallintava suunnittelu lähtee liikkeelle siitä, että Teklaan mallinnetaan rakenteita. Se voi olla useassakin tapauksessa tehokas tapa aloittaa suunnittelu. Kuvassa 4 on esitetty perinteisen suunnitte-

lun ja algoritmiavusteisen suunnittelun välisiä eroja. Kuvasta voidaan todeta, että perinteisen suunnittelun etuna on se, että suunnittelussa saadaan nopeammin aloitettua. Perinteisessä suunnittelussa ei kuitenkaan voida käyttää luonnossuunnitteluun määräänsä enempää aikaa, koska eri vaihtoehtojen vertailu on hidasta. Vaihtoehtoisesti algoritmiavusteisen suunnittelun avulla pystytään luonnosvaiheeseen käyttämään enemmän aikaa, koska algoritmiavusteisesti toteutettuna se on paljon tehokkaampaa. Koska luonnossuunnittelu on pystytty toteuttamaan paljon pitemmälle kuin perinteisellä menetelmällä, on toteutussuunnitteluun ja vaihtoehtojen vertailuun vähemmän aikaa, mutta myös vähemmän tarvetta. Näiden lisäksi vaihtoehtojen tutkiminen algoritmiavusteisesti on paljon nopeampaa ja tehokkaampaa (Woodbury 2010, p. 23-24; Tanska & Österlund 2014, p. 17-18, 24-27).



Kuva 4. *Eri suunnitteluprosessien vertailu (Tanska & Österlund 2014, p. 24).*

Alin osa kuvasta kuvaa tilannetta, jossa algoritmiavusteinen suunnittelu on aloitettu olemassa olevan hyvin samankaltaisen projektin pohjalta. Projektista on pystytty hyödyntämään olemassa oleva algoritmi. Tällä tavalla pyritään hyödyntämään olemassa olevaa tietoa, tai siitä soveltuva osa. Algoritmia muokataan tarvittaessa kyseisen projektin suunnitteluun soveltuvaksi. Tällä tavalla toteutetussa projektissa kyetään saavuttamaan aikataulusäästöjä luonnossuunnitteluvaiheessa. Huomion arvoista on, että suunnittelumenetelmällä ei ole juurikaan vaikutusta toteutusvaiheeseen käytettyyn aikaan. Hyöty ja saatu arvo eivät muodostu pelkästään luonnossuunnittelun, toteutussuunnittelun ja eri vaihtoehtojen vertailuun käytettävästä ajasta, vaan myös niihin käytettävän ajan vaikutuksista. Kun pystytään tehokkaasti käyttämään aikaa eri suunnitteluvaihtoehtojen vertailuun, asiakkaan ja käyttäjien tarpeiden huomioon ottamiseen, on lopputulos mahdollista toteuttaa äärimmäisen monella tavalla. Menetelmän ideana on kuitenkin pystyä löytämään tehokkaammin se ratkaisu, joka on asetetuilta vaatimuksilta, kuten kustannus, tila ja esteettisiltä parempi kuin vastaavasti perinteisen suunnittelumenetelmän avulla saatu (Woodbury 2010, p. 23-24; Tanska & Österlund 2014, p. 17-18, 24-27).

Tavoitteiden määrittäminen suunnittelun lopputulokselle asettaa raamit, joiden perusteella voidaan todeta saavan hyötyä algoritmiavusteisesta suunnittelusta. Jos tarkoituksena on löytää tilojen käytöllisesti paras mahdollinen ratkaisu tai paalulaatan eri osien optimointi, on tavoitteet näillä kahdella täysin erilaiset. Algoritmiavusteisesta suunnittelusta saatava hyöty erilaisissa suunnitteluhankkeissa voidaan saavuttaa monen kaltaisilla menetelmillä. Hyödynnettävien menetelmien käyttö riippuu siitä, mitä tehdään, mitkä ovat tavoitteita, mitä halutaan tehostaa tai optimoida ja onko ohjelmistoissa tai suunnittelijoiden taidoissa rajoitteita (Tanska & Österlund 2014, p. 36).

3.3 Parametrisen mallin luominen algoritmiavusteisesti

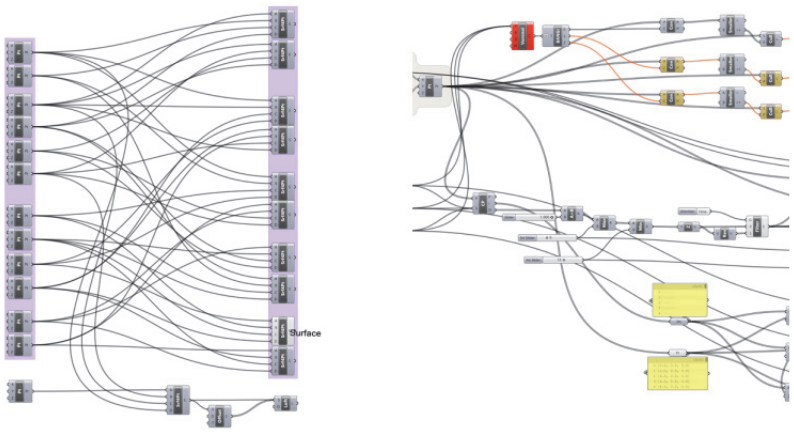
Edellisessä osassa todettiin, että algoritmiavusteisella suunnittelulla pyritään siihen, että jo luotua algoritmia voidaan käyttää lähtökohtana uuden samankaltaisen projektin lähtökohtana. Woodbury toteaa (2010, p. 23-24), että perinteisellä mallintamisella suunnittelun aloittaminen on helppoa ja nopeaa. Hän kuitenkin toteaa myös, että vaikka algoritmiavusteisen suunnittelun aloittaminen on työläämpää, se tuo hyötynsä esiin eri tavalla. Vaikka mallin työläämpi luominen on suunnittelutyölle haaste, on se myös yksi sen tärkeimmistä mahdollisuuksista.

Woodbury esittää (2010, p. 35-38), että tällä hetkellä suunnittelijat mallintavat vain välttämättömällä tasolla. Suunnittelijoiden aloittaessa suunnittelun, he aloittavat aina tyhjältä pöydältä, vaikka he olisivat jo samankaltaisen suunnitelman mallintaneet edellisellä viikolla. Woodbury käyttää tässä yhteydessä termejä rebuild eli uudelleen rakentaminen ja re-use eli uudelleenkäyttö. Hän esittää, että algoritmiavusteisella suunnittelulla on mahdollista siirtyä pois saman asian käsin mallintamisesta siihen, että toistuvat mallintamistyöt voidaan toteuttaa algoritmeilla. Pyrkimyksenä on luoda ja järjestellä algoritmi sillä tavoin, että algoritmia voidaan hyödyntää toisissa projekteissa.

Davis esittää tohtorintyössään (2013, p. 4-6) omasta ja muiden kokemuksiin perustuen osittain eriävän mielipiteen Woodburyn algoritmin uudelleenkäytön ajatusta kohtaan. Hän kertoo olevansa ajatuksen kannalla, mutta kuitenkin havainneensa sekä omassa että muiden parametrisissa malleissa ongelmaksi niiden haurauden. Parametrisen mallin algoritmin muuttaminen ei ole heidän kokemuksien mukaan työmäärältään helppoa, eikä aina edes mahdollista. Suuret algoritmit alkavat olemaan niin monimutkaisia ja epäselviä, että niitä on usein lähes mahdotonta hyödyntää. Woodburyn teoksessa ja Davisin tutkimuksessa annetaan ohjeita parametrisen mallin luomiseen, jotta sen uudelleenkäyttö olisi mahdollista.

Woodbury (2010, p. 24-35) esittää teoksessaan muutamia uusia taitoja, joilla mahdollistetaan algoritmiavusteinen mallintaminen. Ensimmäisenä uutena taitona on ymmärrettävissä olevan datavirran mallintaminen. Hän esittää myös uuden termin kuvaamaan parametrisen mallintamisen luonnetta, ketjusysteemi. Se tarkoittaa datavirran kulkemis-

ta määrättyjen reittien ja pisteiden kautta. Tästä johtuen datavirran mallintamiseen eli ketjusysteemin luomiseen on kiinnitettävä erityistä huomiota.



Kuva 5. *Esimerkki huonosta datavirran mallintamisesta (Davis 2013, p. 130).*

Kuvassa 5 on hyvä esimerkki huonosta datavirran ymmärrettävyydestä. Datavirtaan liittyy olennaisena osana sekä nodejen että isompien kokonaisuuksien riippuvuussuhteiden hallintaa. Mitä enemmän malliin tulee nodeja ja riippuvuussuhteita, sitä hankalammaksi kokonaisuuksien hallinta tulee. Tämän lisäksi mallin muutoskyky heikkenee, koska mallin ymmärrettävyys heikkenee, eikä muutosten aiheuttamista vaikutuksista ole täyttä varmuutta. Tässä osiossa myöhemmin esitetään mittareita koodin ymmärrettävyydelle ja keinoja parantaa datavirran ymmärrettävyyttä.

Toinen Woodburyn esittämä taito on hajauta ja hallitse -tekniikka. Jokainen suunnittelun lopputulos vaatii tietyn määrän tehtäviä ja tehtäväsarjoja. Kun tästä prosessista voidaan helposti erottaa pienempiä kokonaisuuksia, on järkevää käyttää hajauta ja hallitse -tekniikkaa. Tässä tekniikassa suunnittelukokonaisuuden osat suunnitellaan erikseen ja lopulta kasataan yhdeksi algoritmiksi. Tätä tekniikkaa hyödynnetään myös perinteisen koodauksen projekteissa. Kolmas taito on nimeäminen. Periaatteessa äärimmäisen itsestään selvä asia, mutta todellisuudessa vähäksytty taito. Selkeiden nimeämissääntöjen avulla luodaan looginen ja helposti ymmärrettävä kokonaisuus. Nimen pitää olla kuvaava, riittävän selkeä ja yksityiskohtainen, jotta sen avulla voidaan nopeasti päätellä kyseisen osan tehtävä ja sisältö. Nimeämisen lisäksi on hyvä käyttää erilaisia lisätietoa antavia tekstikappaleita (Woodbury 2010, p. 27-29; Davis 2013, 140-141).

Neljäs, viides ja kuudes taito ovat ajattelutapojen muuttamista tai huomioon ottamista suunnittelussa: abstraktisti, matemaattisesti ja algoritmisesti ajatteleminen. Abstraktisti ajattelulla tarkoitetaan tässä yhteydessä sellaisten konseptien luomista, jotka soveltuvat joko muokkaamatta tai muokkaamalla useaan tapaukseen. Jälkimmäinen näistä on kuitenkin parempi kuvaus. On siis tarkoituksen mukaista luoda johonkin tiettyyn kokonaisuuteen soveltuva pohja, jota haluttujen lopputulosten mukaan pystytään tehokkaasti muokkaamaan. Matemaattinen ajattelu on kuitenkin lähtökohta kaikelle mallintamiselle. Piste on koordinaatti avaruudessa, viiva muodostuu kahdesta pisteestä avaruudessa,

jotka yhdistetään piirtämällä viiva pisteiden välille. Algoritminen ajattelu on tapa nitoa nämä kaikki edellä mainitut taidot yhteen. Siinä yhdistyy ajatus prosessista ja täsmällisesti määritetystä tehtäväsarjasta. Algoritmi on täsmällisesti määritetty prosessi, jossa kaiken on oltava oikein määritetty. Yksikin väärin määritetty osa saattaa aiheuttaa algoritmin toimimattomuuden. Algoritmiseen ajatteluun sisältyy olennaisena osana ajatus ohjelmoinnista. Niin algoritmisessa suunnittelussa kuin perinteisessä ohjelmoinnissa on otettava etäisyys konkreettisesta tehtävästä ja ajatella tehtävää erilaisina tehtäväsarjoina, jotta kokonaisuuden hahmottaminen ja suunnittelu ovat mahdollisia toteuttaa (Woodbury 2010, p. 34-35).

3.3.1 Algoritmin toimivuuden arviointi

Algoritmiavusteisen mallin uudelleen hyödyntämiseen kuuluu olennaisena osana sen taipuisuus. Tällä tarkoitetaan mallin toimivuuden säilymistä huolimatta siihen tehtävistä muutoksista. Mallin taipuisuus tai jäykkyys ovat yhteydessä sen tehokkuuteen. Mitä tehokkaammaksi ja taipuisaksi malli halutaan, sitä haastavampaa ja työläämpää se on. Lisäksi on havaittu, että taipuisuus voi olla täysin sattumaan sidonnaista. Tämä johtuu siitä, että mallin herkkyys hajoamiselle on erittäin monen sattuman tai pienen osan summa, jolloin kokonaisuuden ymmärtäminen on haastavaa. Algoritmiavusteisen mallin luomisessa, sen taipuisuudessa ja tehokkuudessa on havaittu selkeitä yhtäläisyyksiä ohjelmistotekniikan kanssa. Osittain pyritään käyttämään ohjelmistotekniikan puolella hyväksi havaittuja keinoja tutkia ja testata koodin taipuisuutta (Davis 2013, p. 5-7, 195, 197).

Davisin (2013, p. 75-91, 195-201) tohtorintyössä esitetään hyviksi keinoiksi mitata algoritmiavusteisen mallin koodin toimivuutta:

- Määrälliset kriteerit
    - Koodin rivimäärä
    - Koodin kompleksisuus
    - Luomiseen käytetty aika
    - Muokkaamiseen käytetty aika
    - Viive
    - Syötteiden määrä
- Laadulliset kriteerit
    - Oikeellisuus
    - Taipuisuus
        - Kyky mukautua muutoksiin
    - Uudelleenkäyttämisen mahdollisuus
    - Tehokkuus
    - Helppokäyttöisyys
    - Toiminnallisuus

Näitä kriteereitä käytetään tyypillisesti myös ohjelmistotekniikassa. Davis käytti näitä kriteereitä tutkimuksessaan, joiden avulla hän arvioi työnsä kolmessa eri case-tutkimuksessa algoritmiavusteisten mallien koodia. Samalla nämä mittarit toimivat suunnittelijan ajatustyön apuvälineinä. Niiden avulla suunnittelija pystyy jäsentelemään suunnittelua ja muuttamaan ajattelutapaa algoritmiavusteisen suunnittelun vaatimalla tavalla (Davis 2013, p. 75-91, 195-201).

Koodin rivimäärän käyttäminen mittarina on ristiriitaista. Sen avulla voidaan todeta koodin virheiden ja rivien lukumäärän välille suora yhteys. Tämän lisäksi rivien lukumäärällä on myös suora yhteys koodin kompleksisuuteen. Algoritmiavusteisessa mallintamisessa koodin rivimäärä voidaan ajatella vastaavan nodejen määrää. Node tarkoittaa solmupistettä tai komponenttia algoritmissa. Näistä huolimatta, algoritmi voi olla joko pitkä tai lyhyt ja toimia juuri halutulla tavalla. Koodin koko on siis hankala mittari, koska se ei ole millään määrin yksiselitteinen. Koodin kompleksisuudella tarkastellaan koodin rakennetta ja jäsentelyä. Yksinkertaisuudessa sen voidaan ajatella kuvaavan reittien lukumäärää, joita pitkin data virtaa. Tämä ei kuitenkaan ole riittävän kattava selitys. Paremminkin se voidaan ajatella kuvaavan aikaa ja työtä, joka vaaditaan koodin ymmärtämiseen, joka perehtyy koodiin ensimmäistä kertaa (Davis 2013, p. 75-87).

Mallin luomiseen käytetty aika kuvastaa nimensä mukaisesti aikaa, joka vaaditaan mallin luomiseen tyhjästä. Mittauskriteerinä se ei välttämättä ole paras, sillä mallin luomiseen vaadittu aika on hyvin riippuvainen sen tekijän taidoista. Muokkaamiseen käytettävä aika on parempi kriteeri. Algoritmin muokkaaminen ja uudelleen käyttö on yksi tärkeimmistä hyödyistä, joita algoritmiavusteiselle mallintamiselle esitetään. Muokkaamisella tarkoitetaan sekä parametristen arvojen, algoritmin tai sen osakokonaisuuksien muokkaamista. Tämä ei ole eksakti arvo, joka voidaan osoittaa, koska aika on hyvin riippuvainen sen muokkaajan taidoista ja tottumuksista. Muokkaamiseen käytettävää aikaa voidaan kuitenkin käyttää suuntaa antavana arvona. Viive on aika, joka menee algoritmin suorittamiseen. Se on siis riippuvainen koodin riveistä, kompleksisuudesta ja tietokoneen laskentatehosta (Davis 2013, p. 75-87).

Mallin oikeellisuus on oikeastaan lähtökohtana mallin luomiselle. Mallin on tuotettava haluttu lopputulos. Sen on myös toimittava halutulla tavalla. Tätä voidaan pitää hyvinkin oleellisena kriteerinä. Taipuisuus kuvastaa mallin kykyä sopeutua muutoksiin. Taipuisuuden voidaan todeta olevan suorassa yhteydessä koodin rivien määrään ja kompleksisuuteen. Uudelleen käyttämisen mittari kuvastaa algoritmiavusteisen mallin hyödyntämistä joko osissa tai kokonaisuutena. Tehokkuutta voidaan kuvata yksinkertaisesti mallin aiheuttamalla paineella tietokoneen suorittimelle. Tehokkuudella on myös yhteys mallin käyttötarkoitukseen ja algoritmin määrittämisestä liian tarkasti yhteen käyttötarkoitukseen. Tällä tarkoitetaan sitä, että mitä tarkemmaksi mallin käyttötarkoitus tehdään, sitä vähemmän sitä voidaan uudelleen käyttää (Davis 2013, p. 88-91).

Helppokäyttöisyys tarkoittaa yksinkertaisesti sitä, että algoritmista mallia on helppo käyttää. Helppokäyttöisyys on hyvin riippuvainen myös muista kriteereistä, kuten mallin kompleksisuudesta. Tämän avulla oikeastaan määräytyy mallin luomiseen ja muokkaamiseen käytetty aika. Kuten on jo mainittu, ne ovat kuitenkin hyvin riippuvaisia käyttäjän taidoista ja tottumuksesta malliin. Toiminnallisuus kuvastaa mallin avulla suoritettavia tehtäviä tai mahdollisuuksia ja niiden lukumäärää. Laadulliset kriteerit ovat kohtuullisen hankalia mitata tai todeta. Osa voidaan kohtuullisen helposti todeta jonkin menetelmän avulla tai ammattilaisen toteamana. Laadullisten kriteereiden avulla pystytään luomaan hyvä sanasto ja pohja, joiden avulla pystytään paremmin tutkimaan algoritmisten mallien ominaisuuksia (Davis 2013, p. 75-87).

3.3.2 Algoritmin ryhmittely ja siitä saatava hyöty

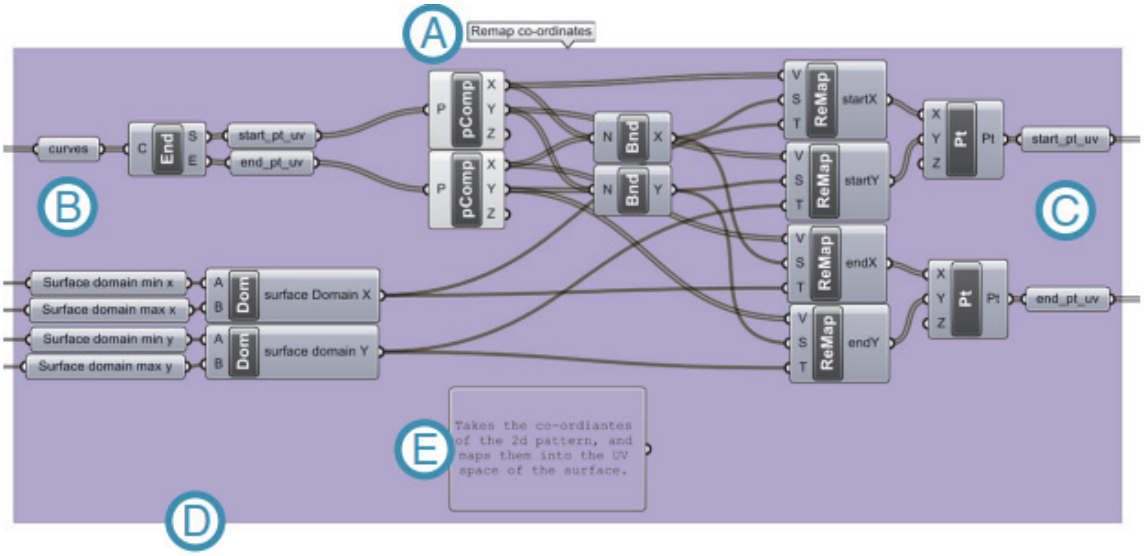
Algoritmi on tehtäväsarja, jonka jokaisella osalla on oma tehtävänsä ketjussa. Onkin loogista hajottaa nämä tehtäväsarjat, jotta jokainen tehtävä voidaan yksitellen suunnitella ja toteuttaa. On kyse hajauta ja hallitse -tekniikasta, joka esiteltiin työssä jo aikaisemmin (Woodbury 2010, p. 27-28). Myöhemmin Woodbury esittää tekniikan pohjalta muodostuvan käsitteen moduuli. Moduuli on rajattu tehtäväsarja, jolla on aina sekä sisään tuleva että ulos menevä datavirta. Käytännössä se tarkoittaa juurikin algoritmin yhtä tehtäväsarjaa, mutta se voidaan käsittää myös hieman isompana kokonaisuutena. Ideana moduulilla on, että se voi olla täysin itsenäinen osa algoritmista mallia, ja sillä on täysin oma tarkoituksensa siinä. Moduulien avulla pystytään vähentämään koodin kompleksisuutta ja lisäämään sen uudelleenkäytön mahdollisuutta (Woodbury 2010, p. 45-46).

Davis (2013, p. 127-153) tukee tätä ajatusta moduuleista. Työssään hän perehtyy moduuleihin ja yleisesti algoritmiavusteiseen suunnitteluun. Algoritmiavusteisessa suunnittelussa hän keskittyy mallin luomisen logiikan tärkeyteen. Lisäksi hän tutkii, miten tuo logiikka voidaan saavuttaa.

Kuvassa 6 Davis esittää kuinka Grasshopper-ohjelmassa moduulit esitetään. Moduulien luomiseen käytetään Grasshopperin omaa työkalua group (ryhmä). Kuvan kirjainten selitykset:

- A
    - Moduulin nimi
- B
    - Datan sisääntulo
- C
    - Datan ulosmeno
- D
    - Moduuli suljetaan laatikon ulkoreunoilla, jolloin data kulkee sisälle ainoastaan kohdasta B ja ulos kohdasta C

- E
    - Tarkka kuvaus moduulin tehtäväsarjasta



Kuva 6. *Moduuli Grasshopper -ohjelmassa (Davis 2013, p. 128).*

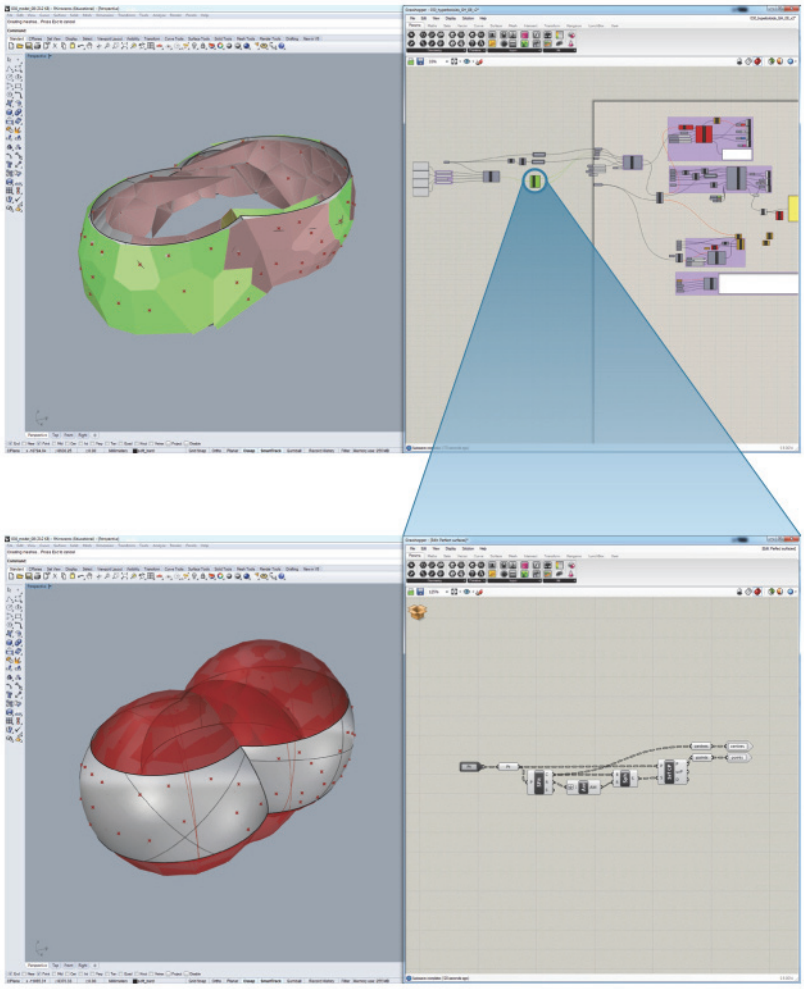
Davis esittää (2013, p. 127-129) oikein määritettyjen ryhmittelyiden ja niiden käyttämisestä saatavia hyötyjä:

- Hajaannuttaminen
    - Monimutkaiset ongelmat voidaan jakaa useaan erilliseen tehtäväsarjaan, jotka sisältävät kyseiseen ongelmaan liittyviä pienempiä ongelmia. Tällä tavalla työskennellessä suunnittelijat voivat samaan aikaan ratkoa yhteisen ongelman eri osa-alueita.
- Kokoonpaneminen
    - Parhaassa tilanteessa moduulit saadaan luotua täysin itsenäisiksi, jolloin niihin ei tarvitse kuin syöttää data. Tällöin päästään tilanteeseen, joissa erillisiä moduuleita voidaan tehtävän luonteesta riippuen kytkeä halutulla tavalla toisiinsa. Tämä mahdollistaa koodin uudelleenkäyttämisen parhaalla mahdollisella tavalla.
- Ymmärrettävyys
    - Kun moduulit luodaan oikein, niissä on selitykset ja loogiset datavirrat, ovat ne helpoiten ymmärrettävissä. Tällä tavalla pystytään vähentämään ylimääräistä aikaa, joka kuluu mallin toiminnan ymmärtämiseen.
- Muutoskyky
    - Moduuli on muutoskykyinen ja taipuisa, kun siihen pystytään tekemään muutoksia kohtuullisella työmäärällä suhteutettuna saatuun hyötyyn. Periaatteessa työmäärää ja siihen kuluvaa aikaa verrataan aikaan, joka kuluisi koodin tyhjästä luomiseen.
- Haavoittumattomuus

- Jokainen moduuli tulee pystyä testaamaan ja löytämään virheellinen koodi. Tämä tulee olla mahdollista ilman koko mallin rikkomista.

Woodbury (2010, p. 27-28) esittää teoksessaan parametriseen mallintamiseen menetelmän ”hajauta ja hallitse”. Samoin Davis esittää, että tiettyyn lopputulokseen pääsemiseksi, on järkevää hajauttaa tehtäväsarja erillisiin kokonaisuuksiin. Woodbury esittää myös samoja hyötyjä, joita Davis listaa modulaarisesta työskentelystä.

Grasshopper sisältää modulaarista työskentelyä helpottavan työkalun cluster. Sen avulla moduulien sisältö piilotetaan, jonka lopputuloksena on vain yksi node, jossa on samat datan sisään- ja ulostulot, kuin alkuperäisessä moduulissa. Clusterin saa avattua, jolloin pystyy tarkastelemaan ja muokkaamaan moduulin sisältöä. Se tulee myös nimetä erittäin loogisesti, jotta on selkeää, mitä kyseinen cluster tekee (Tedeschi 2011, p. 113-115; Davis 2013, p. 130-132).



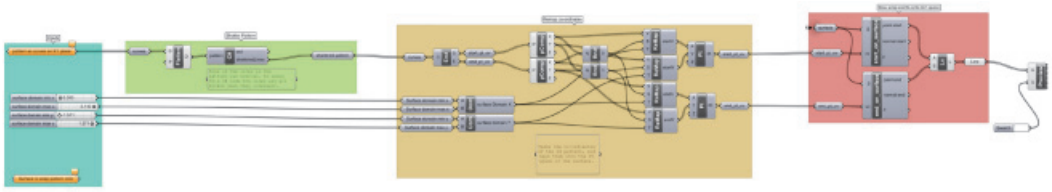
Kuva 7. *Grasshopperin cluster-työkalu (Davis 2013, p. 131).*

Kuvassa 7 on esitetty miltä cluster näyttää koko työn canvaksella. Alemmassa kuvassa on cluster avattu, jolloin pystytään tarkastelemaan clusterin sisältöä. Suurimmat hyödyt

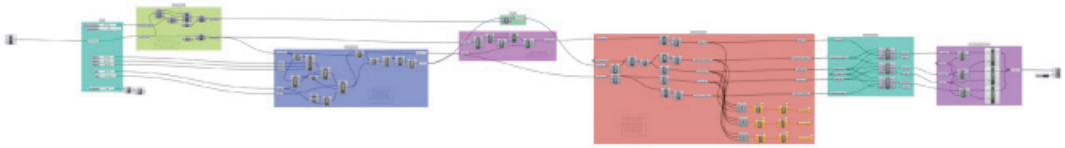
clusterista on canvaksen selkeytyminen, kun nodejen ja wirejen määrä vähenee merkittävästi (Tedeschi 2011, p. 113-115; Davis 2013, p. 130-132).

3.3.3 Algoritmiavusteisen mallin jäsentely ja selkeys

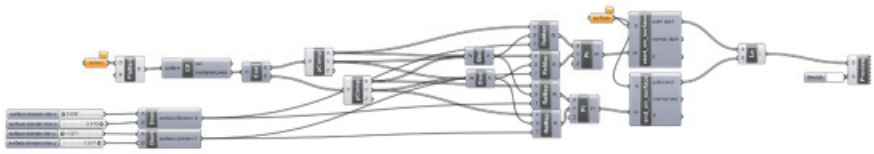
Davis teki työssään (Davis 2013, p. 136-142, 142-149, 152-153) kolme erillistä tutkimusta, joista kahteen perehdytään tässä työssä. Ensimmäisessä tutkimuksessa hän tutki kuinka arkkitehtiopiskelijat arvioivat Grasshopperilla tehtyjen algoritmien selkeyttä. Testiin valittiin 25 oppilaan joukosta 4 opiskelijaa. Heillä oli 1-7 vuoden kokemus tietokoneavusteisesta suunnittelusta (CAD) ja 1 vuoden kokemus Grasshopperin käytöstä. Tutkimuksessa heille esitettiin kuvassa 8 olevat 3 eri algoritmia.



Model-A1
Nodes: 41
Structured: Yes
Function: Wraps two-dimensional pattern onto a surface
Equivalent to: Model-C1



Model-B
Nodes: 120
Structured: Yes
Function: Draws triangles on a hemisphere from an inscribed polyhedron
Equivalent to: n/a



Model-C1
Nodes: 26
Structured: No
Function: Wraps two-dimensional pattern onto a surface
Equivalent to: Model-A1

Kuva 8. *Davisin ensimmäisessä testissä esittämät parametriset mallit (Davis 2013, p. 137).*

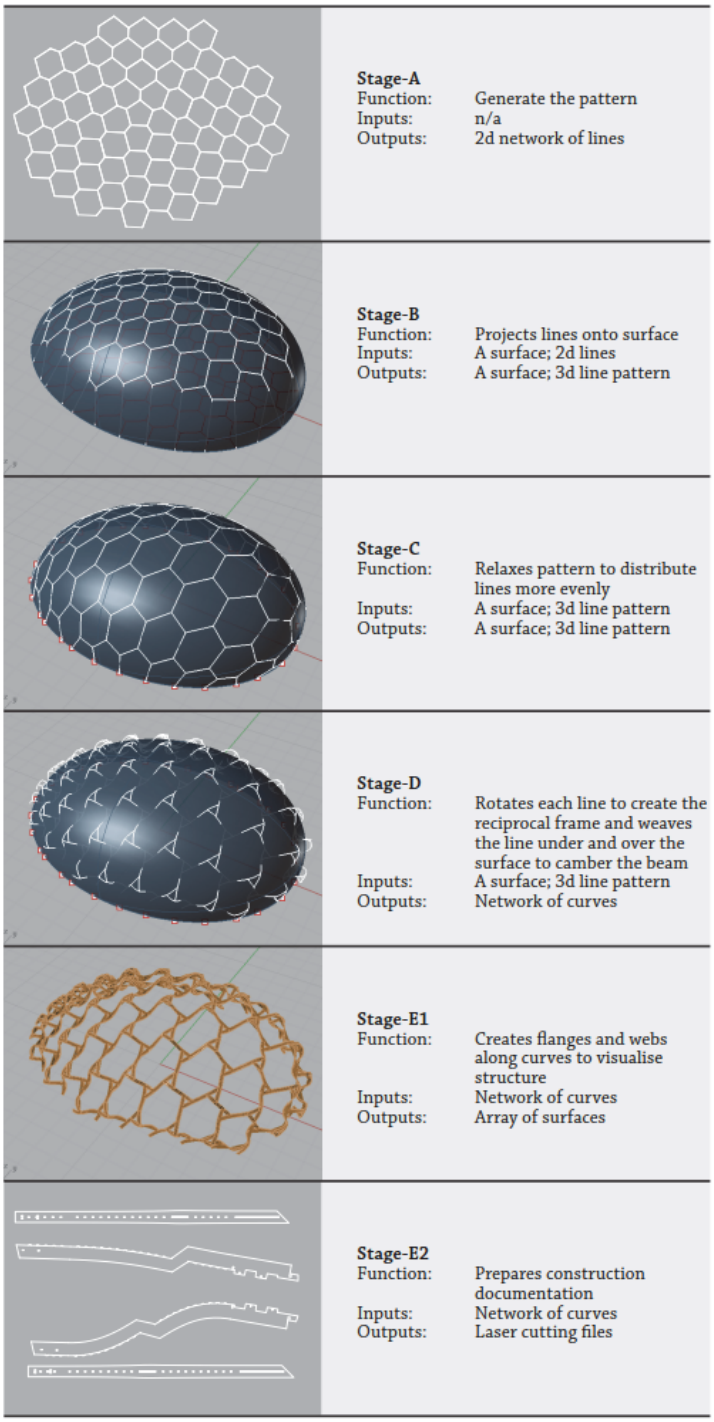
Model-A1 ja Model-C1 ovat täsmälleen samoja malleja, mutta A1 on loogisesti ryhmitelty ja järjestetty, kun taas C1 ei ole. Model B oli testissä mukana, jotta osallistujat eivät päättelisi kahden mallin olevan samoja. Osallistujien tehtävänä oli kuvailla kuinka mallin imputit ja nodet kontrolloivat luotavaa geometriaa. Heidän piti samalla ääneen selittää heidän päättelyketjua. Tutkimuksesta Davis sai tulokseksi 4 avaintekijää, jotka auttoivat tutkimukseen osallistujien algoritmien hahmottamista:

- Nimeäminen
- Sijoittelu

- Selitykset
- Ryhmittely

Toinen tutkimuksista oli laajempi. Tutkimus jakaantui kolmeen osaan: ensimmäisessä osassa työryhmä teki algoritmiavusteisesti tarkoituksellisesti puutteellisten lähtötietojen pohjalta erilaisia jäsentelemättömiä parametrisia malleja; toisen osan aluksi workshopissa valittiin algoritmi, jolla suunnittelua jatkettiin, jonka jälkeen Davis alkoi järjestellä algoritmia ymmärrettävämmäksi edellä esitettyjen kriteerien pohjalta; kolmannessa vaiheessa työryhmälle annettiin tarkat tiedot lopullisesta rakennuksesta, johon oli tehty tarkoituksen mukaisesti suuria muutoksia. Ensimmäisen vaiheen tarkoituksena oli lähtötietojen pohjalta luoda mahdollisimman taipuisa algoritmi, jota voidaan muokata, kun loput suunnittelutiedot saadaan. Puuttuvia tietoja oli muun muassa rakennuksen pinnan muoto, palkkien liitosdetaljit ja kokonaisuudessaan rakennuksen rakenteellinen toimivuus. Vaikka merkittävä osa lähtötiedoista puuttui, heillä oli näistä lähtötiedoista tiettyjä tietoja kuitenkin olemassa, kuten että rakennuksen pinnan tuli olla kaksoiskaareva.

Toisessa osassa Davis alkoi jäsennellä valittua algoritmia moduuleihin ja selkeisiin kokonaisuuksiin. Tarkoituksena oli testata hänen teoriaansa algoritmin jäsentelyn ja logiikan vaikutuksista sen ymmärrettävyyteen, taipuisuuteen ja algoritmin muuttamiseen kuluvasta ajasta. Kuvassa 9 esitetään Davisin luomat moduulit ja selitetään niiden sisältö.



Kuva 9. *Davisin luomat moduulit (Davis 2013, p. 145).*

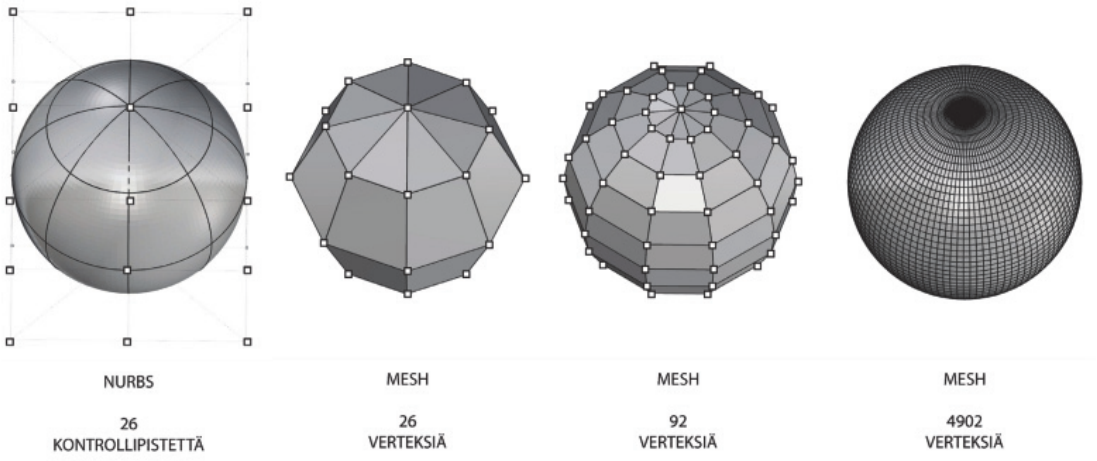
Kolmannessa vaiheessa kun algoritmi oli jäsennelty, työryhmä otti projektin täysin haltuunsa. Davis toimi tarkkailijana, jossa tavoitteena oli selvittää kuinka mallia tuntemattomat suunnittelijat kykenevät sisäistämään mallin logiikan ja kykenevät tekemään siihen merkittäviä muutoksia. Heidän tehtävään oli tehdä kolme isoa muutosta malliin. Ensimmäinen muutos onnistui todella helposti, mutta kaksi muuta vaativat suunnittelijoilta enemmän työtä, joka oli myös tarkoituksellista. Lopputulemana suunnittelijat olivat yhdistäneet ja hyödyntäneet Davisin luomia moduuleita onnistuneesti. Suunnitteli-

joita ei oltu neuvottu lähes millään tavalla, miten päästä haluttuun lopputulokseen, vaan suunnittelijan oli itse tutkittava ja ymmärrettävä ja sen jokaisen moduulin tehtävät.

Onnistuneesta tuloksesta pääteltiin, että loogisesti toteutettu moduuleihin jako, mallin ryhmittely ja nimeäminen mahdollistivat täysin mallille tuntemattoman suunnittelijan kykenevän omaksumaan suuren mallin sisällön ja soveltamaan sitä. Tuloksista voidaan todeta mallin taipuisuuden vaikuttavan positiivisesti mallin muutoskykyyn. Muutoskyky vastaavasti on hyvin olennainen osa sekä mallin hyödynnettävyyttä että sen arvoa, sillä voidaan todeta muutoskyvyn vastaavan muutoksiin kuluvaa aikaa ja siten mallista saatavaa arvoa. Jos mallia kykenee muokkaamaan ainoastaan sen luonut henkilö, mallista saatava arvo on pahimmassa tapauksessa hyvinkin lyhytikäinen. Mallin uudelleenkäyttö tuo aina lisäarvoa mallille, koska tällä tavalla saadaan erilaisia säästöjä aikaiseksi. Näistä tutkimuksista voidaan tehdä yksinkertainen yhteenveto, että mallin muutoskykyä, ymmärrettävyyttä, jatkuvuutta ja mallin uudelleenkäyttöä voidaan edistää selkeällä ja johdonmukaisella nimeämisellä ja ryhmittelyllä.

3.4 Grasshopper ja Rhinoceros 3D

Diplomityössä hyödynnetään McNeel yhtiön tuotetta Rhinoceros 3D ja David Ruttenin kehittämää lisäosaa Grasshopperia. Rhino on monimuotoisten pintojen ja viivojen kolmiulotteinen mallinnusohjelmisto. Pinnat ja viivat perustuvat NURBS-kappaleista (Non-Uniform Rational B-Splines). NURBS-muotojen avulla pystytään määrittämään mesh-pohjaisia malleja tarkemmin kappaleiden kolmiulotteinen geometria. Kuvassa 10 on selkeästi esitetty millä tavalla muissa ohjelmistoissa pintojen, varsinkin kaarevien pintojen mallintaminen tapahtuu. Vaikka nykytietokoneissa laskentateho on korkea, niin on melko selkeää, että 26 kontrollipisteen hallinta on helpompaa ja tehokkaampaa, kuin 4902 pisteen (What are NURBS?; Tanska & Österlund 2014, p. 28-31).

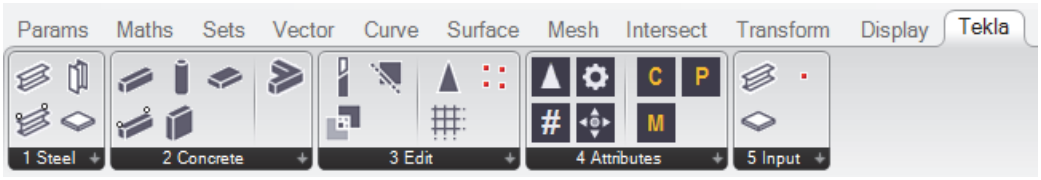


Kuva 10. *NURBS-pinnoista ja MESH-verkoista muodostettu ympyrä (Tanska & Österlund 2014, p. 30).*

Rhino on periaatteessa yksinkertainen, mutta ammattikäyttöön soveltuva mallinnusohjelmisto, johon löytyy paljon käyttökelpoisia lisäosia. Lisäosilla työkalusta saadaan soveltuva käyttäjän omiin tarpeisiin. Työkalua voidaan kehittää lisäosilla soveltumaan esimerkiksi erilaisiin päivänvalo- ja rakenneanalyyseihin tai algoritmiavusteiseen mallintamiseen. Rhinon työkaluja voi myös itse ohjelmoida käyttämällä Rhinoscript tai Python -ohjelmointikieliä (Tanska & Österlund 2014, p. 28-31).

Grasshopper on visuaalista skriptausta hyödyntävä lisäosa Rhinoon. Sen avulla luodaan algoritmeja, joilla ohjataan Rhinon mallintamista. Grasshopperin suurin etu lienee visuaalinen koodaus, joka madaltaa algoritmiavusteisen suunnittelun kynnystä huomattavasti. Ohjelmisto sisältää erilaisia valmiita ohjelmakomponentteja, joita linkittämällä luodaan algoritmi. Algoritmin tarkoituksena on tuottaa haluttu kappale Rhinoon tai erilaisten lisäosien avulle toisiin suunnitteluohjelmiin. Algoritmin kaikilla eri ohjelmakomponenteilla on oma tarkoituksensa. Ohjelmakomponentteja on useita erilaisia, riippuen käyttötarkoituksesta. Valmiiden ohjelmakomponenttien lisäksi voi käyttää yhteisön tai yritysten kehittämiä ohjelmakomponentteja. Vaihtoehtoisesti ohjelmakomponentteja voi myös itse koodata käyttäen perinteistä tekstipohjaista koodia. Koodikielenä toimii VB.net, C# ja Python (Tanska & Österlund 2014, p. 28-31).

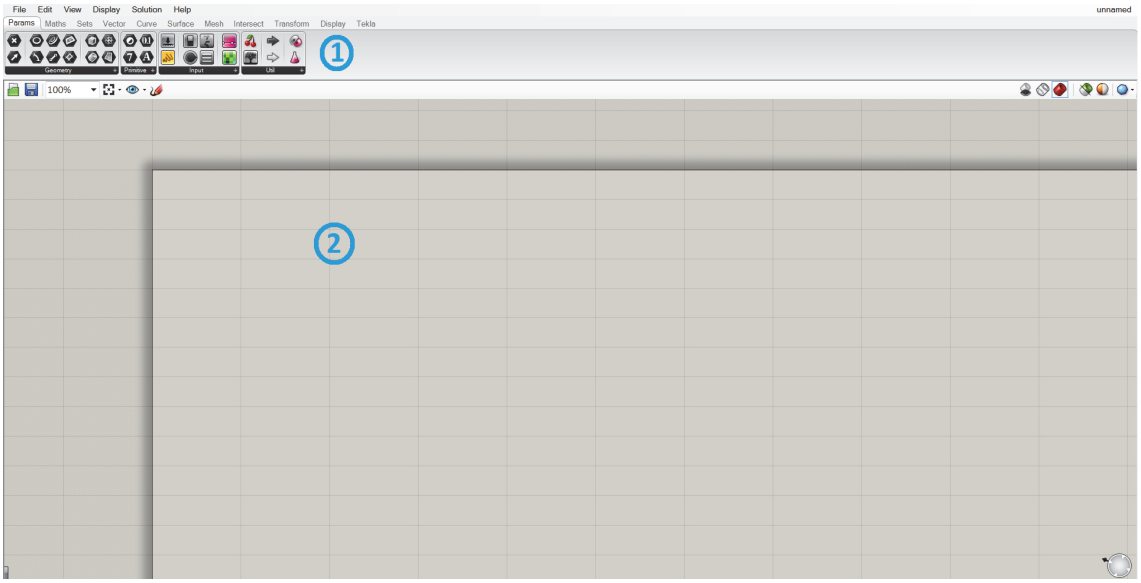
Trimble Solutions Corporation on kehittänyt Grasshopperin ja Tekla Structures -ohjelmistojen välille automaattisen linkin. Tämän linkin avulla Teklaa voidaan komentaa Grasshopperin avulla (Grasshopper-Tekla Live Link).



Kuva 11. *Grasshopper-Tekla Live Link -lisäosa*

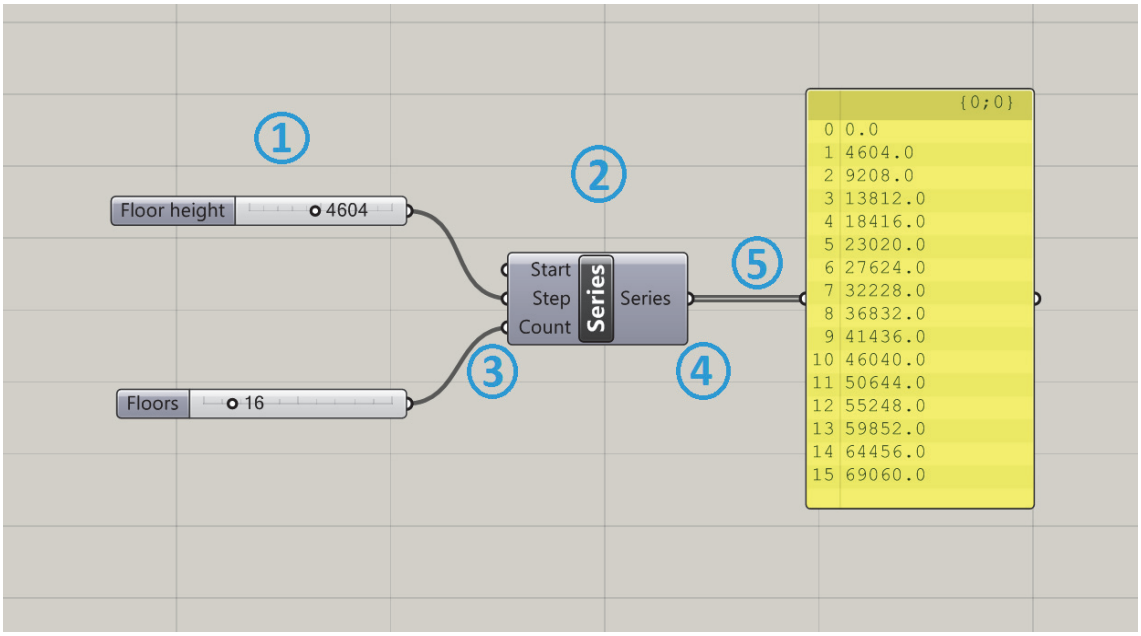
Lisäosan ladattua Grasshopperin komponenttipalkkiin ilmestyy Tekla-niminen välilehti kuvan 11 mukaisesti. Välilehden alla on muutamia Teklan työkaluja. Tätä linkkiä tullaan käyttämään olennaisena osana diplomityön case-tutkimusta. Linkkiin ja sen toimintaan perehdytään tarkemmin case-tutkimuksessa.

Seuraavaksi selitetään lyhyesti Grasshopperin käyttöliittymä ja termistö. Työssä päätettiin käyttää englanninkielisiä termejä, jotta vältytään käännösten aiheuttamista epäselvyyksistä. Lisäksi, puhekielessä useat englanninkieliset termit kääntyvät suoraan sellaisena suomenkieliseksi.



Kuva 12. *Grasshopper käyttöliittymä.*

Kuvassa 12 numero 1 alue on component menu. Kaikki Grasshopperin työkalut löytyvät component menun välilehdiltä. Numero 2 alue on canvas, johon koodi eli algoritmit kirjoitetaan.



Kuva 13. *Grasshopperin peruskäsitteet.*

Kuvassa 13 esitellään Grasshopperin peruskäsitteet. Numero 1 kuvastaa parametreja, eli arvoja. Numero 2 kuvastaa komponenttia tai nodea. Jokainen komponentti on omanlainen tehtäväsarja, jonka taustalla on koodia, joka ohjaa komponentin toimintaa. Numero 3 kuvastaa kyseisen komponentin inputteja, eli datan sisääntuloja. Numero 4 kuvastaa komponentin outputtia, eli datan ulosmenoa. Numero 5 kuvastaa komponenttien yhdistämistä wire:illä tai string:eillä. Data virtaa niiden avulla komponentista tai parametrista toiseen. Tässä lyhyessä algoritmissa luodaan sarja, jossa kerrosten lukumäärä on 16 ja kerroskorkeus 4604 mm.

3.5 Tämän hetken sovellutukset alalla

Algoritmiavusteisesta suunnittelua hyödyntäviä menetelmiä on olemassa useita erilaisia. Arkkitehdit hyödyntävät luontoa ja evoluutiota erilaisten muotojen ja vaihtoehtojen kartoittamiseen. On olemassa erilaisia luonnon käyttäytymis- ja kasvumalleja, joiden avulla kyetään imitoimaan luonnon tapaa käyttäytyä ja kehittyä. Rakennesuunnittelussa yhtä lailla voidaan hyödyntää edellisiä menetelmiä. Evolutiivisilla menetelmillä kyetään optimoimaan rakenteita. Erilaisten fysiikkasimulaattoreiden, analyysityökalujen ja tiedonsiirron avulla mahdollistetaan FEM-analyysi algoritmisesta mallista.

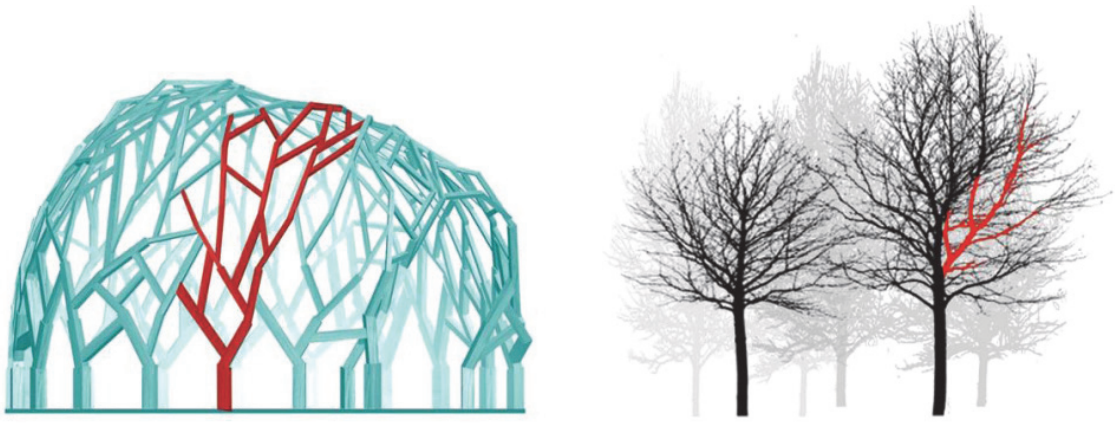
3.5.1 Luonto ja evoluutio suunnittelun apuna

Arkkitehdit ovat omaksuneet algoritmiavusteisen suunnittelun nopeimmin. Sen avulla arkkitehdit ovat pystyneet luomaan uusia menetelmiä tukemaan luovaa suunnittelutyötään ja poistamaan toistuvia tehtäviä. Morfogeneesi ja biomimettiikka ovat esimerkkejä uusista menetelmistä Biomimetiikka terminä tarkoittaa luonnon menetelmien ja sääntö-

jen tutkimista, sekä niiden hyödyntämistä teknisissä sovelluksissa. Morfogeneesi on yksi biomimetiikan sovellutuksista. Morfogeneesi tarkoittaa organismin muotojen ja rakenteiden syntymekanismia. Ne pohjautuvat yksinkertaisiin luonnon sääntöihin, joita pyritään jäljentämään (Tanska & Österlund 2014, p. 20).

Biomimetiikan avulla luodaan biologiasta saatuja muotoja, rakenteita ja prosesseja. Luonto on evoluution avulla tehnyt omaa rakenteiden ja muotojen optimointia, jolloin on järkevää pyrkiä hyödyntämään niitä. Kun biomimetiikkaa halutaan hyödyntää rakenteiden suunnittelussa, on määritettävä rakenteen toiminnalliset tehtävät. Ne liittyvät yleensä rakenteen muotoon ja ominaisuuksiin, mutta tehtävät on ajateltava yhtenä kokonaisuutena, jotta biomimetiikan avulla voidaan jäljennellä luonnon tapaa käsitellä ja ratkoa ongelmia. Hyvänä esimerkkinä luonnon käyttäytymisen mallintamisesta on lintujen parviälykkyys. Algoritmi perustuu kahteen asiaan, yksittäisten lintujen liikkeeseen ja parven liikkeeseen. Yksittäisten lintujen liikkeen muutokset vaikuttavat muiden lintujen liikkeisiin, jolloin päästään ns. hajautetun älykkyyden päätöksiin (Tanska & Österlund 2014, p. 44-47).

Toinen mielenkiintoinen morfogeneesin menetelmä on L-systeemi. Sillä pyritään simuloimaan kasvien kasvua, alun perin levien kasvamista, mutta myöhemmin havaittu sen soveltuvan myös muiden kasvien kasvun jäljittelyyn. Yksinkertaisuudessaan systeemi perustuu kasvusääntöihin, joissa kaavan monistuminen ja muuttuminen syntyvät iteroimalla edelliseen kasvutapahtumaan. Kuvassa 14 on hyvä esimerkki L-systeemistä. Sen on suunnitellut vuonna 2009 Lundén Österlund arkkitehdit. Rakennuksen rungon on pyritty mallintamaan puun kasvua L-systeemin avulla. Runko koostuu 19 puusta, jotka itsenäisesti kasvavat L-systeemin mukaisesti, ja lopulta puiden latvusto rakentuu yhtenäiseksi rakenteeksi (Tanska & Österlund 2014, p. 44-47, 100).

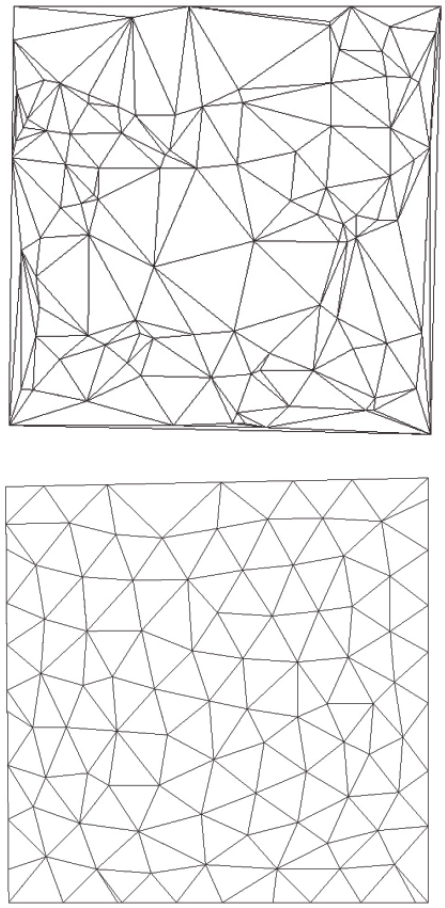


Kuva 14. *Ligna-paviljongi on mallinnettu L-systeemin avulla (Tanska & Österlund 2014, p. 47).*

Evolutiiviset menetelmät hyödyntävät luonnossa esiintyvää evoluutiota erilaisissa optimointi- ja ongelmanratkaisutilanteissa. Evoluution perusta on luonnon valinnassa, joka tarkoittaa yksilön periytyvien ominaisuuksien muovautumista ympäristön mukaan.

Tyypilliset ominaisuudet ovat sukupolvittain etenevä kehitys, ominaisuuksien periytyminen, erilaisista yksilöistä muodostuva populaatio ja yksilöiden erikoistuminen mutaation kautta. Menetelmä jakaantuu kahteen osioon, genotyyppiin eli koodattuun rakennusprosessin tietoon kaikissa yksilöissä ja fenotyyppiin, joka on ympäristöolosuhteiden ja genotyypin lopputulema. Fenotyypissä yksilöön koodattu rakentamisprosessi muokkautuu ympäristöolosuhteiden mukaisesti, jolloin lopputulemana on täysin ainutlaatuinen yksilö (Tanska & Österlund 2014).

Kun evoluutiota hyödynnetään optimointimenetelmässä, kyseessä on luonnonvalintaa hyödyntävä heuristinen menetelmä. Siinä optimaalista tulosta haetaan hakuavaruudesta. Yleensä suurinta hyötyä tästä menetelmästä saadaan, kun ongelmassa on kyse monen muuttujan tilanteesta. Paras ratkaisu valikoituu ns. hyvyysfunktion avulla, joka on matemaattinen kaava, jolla arvotetaan haluttu ratkaisu. Ratkaisuja voidaan hakea myös mutaation avulla. Haasteeksi muodostuu hyvyysfunktion määrittäminen ja laskenta-aika. Tietokone laskee ja pisteyttää hyvyysfunktion avulla hakuavaruuden ratkaisuja. Mitä pitempi laskenta-aika on, sitä lähemmäksi ns. oikeaa eli optimaalisinta ratkaisua päästään (Makris 2013, p. 98-106; Tanska & Österlund 2014, p. 48-49).



Kuva 15. *Geneettisen algoritmin avulla tehty kolmioverkon optimointi (Tanska & Österlund 2014, p. 49).*

Kuvassa 15 esitellään selkeä esimerkki geneettisen algoritmin hyödyntämisestä optimoinnissa. Ylhäällä on satunnaisesta pistejoukosta luotu kolmioverkko ja alhaalla optimoinnin tulos. Optimoinnin reunaehtoina oli, että jokaisen viivan pituus olisi mahdollisimman lähellä kaikkien pituuksien keskiarvoa. Optimoinnin lopputilanne on 3688 sukupolven jälkeen (Tanska & Österlund 2014, p. 49).

Grasshopperiin on integroitu oma optimointityökalu Galapagos. Galapagos hyödyntää evolutiivisia menetelmiä optimoinnissa. Yksinkertaisuudessaan optimointi tapahtuu määräämällä muuttujat, muuttujille rajat ja vaihtoehtojen määrä eli muuttujien askelten välit. Näiden lisäksi käyttäjä määrittää fitness variable eli vapaasti suomennettuna määräävät laadulliset ominaisuudet, eli hyvyysfunktio. Niiden avulla Galapagos kykenee arvioimaan kaikkien mahdollisten ratkaisuiden laatua ja pisteyttämään ne, jotta voidaan löytää paras mahdollinen ratkaisu. Rakenteiden optimointiin perehdytään tarkemmin tämän osion lopussa (Makris 2013, p. 49-51).

3.5.2 Algoritmiavusteinen suunnittelu rakennesuunnittelussa

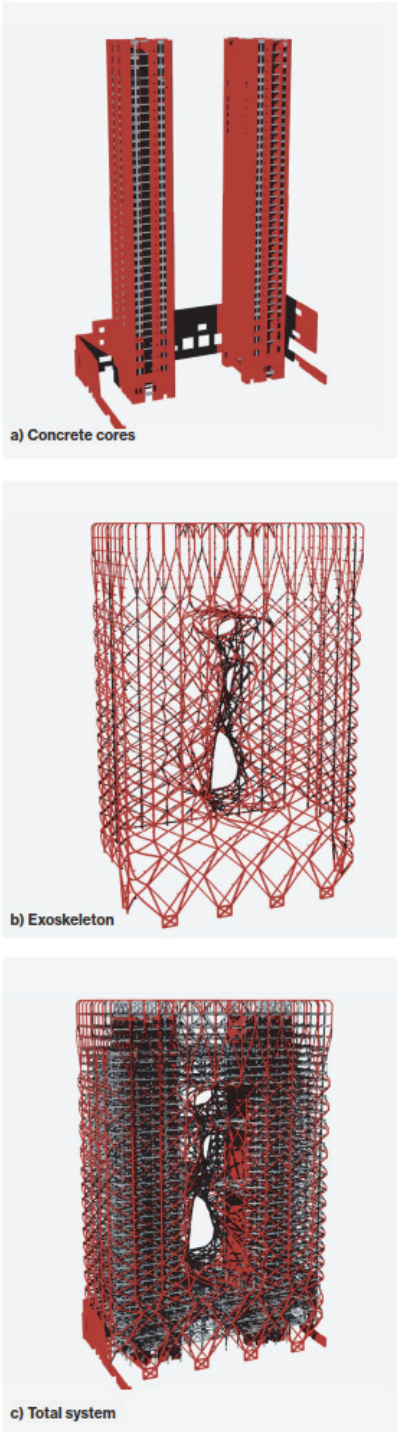
Algoritmiavusteista suunnittelua käytetään rakennesuunnittelussa vähemmän mitä arkkitehtien keskuudessa. Tutkimuksia aiheesta on myös suhteellisen vähän. Eero Särkkä

teki 2015 vuonna opinnäytetyönsä aiheesta ”Paalulaatan suunnittelu parametrisen tietomallin avulla”. Työssään hän tutustuu lyhyesti parametrisen mallin algoritmiavusteisen suunnittelun taustoihin, esittelee käytettävät ohjelmistot ja mallintaa paalulaatan suunnitteluprosessin uudella algoritmiavusteisella suunnittelumenetelmällä. Algoritmiavusteinen suunnittelu alkoi olemassa olevan suunnitteluprosessin pilkkomisesta rajattuihin selkeisiin kokonaisuuksiin. Kun prosessi oli jaettu osiin, pystyttiin algoritmi luoda osio kerrallaan. Lopputuloksena Särkkä koki algoritmiavusteisen suunnittelun mahdollisuudet hyviksi. Hän korostaa, että alalla tarvittaisiin enemmän uskallusta lähteä kehittämään ja kokeilemaan uusia menetelmiä. Hänen mielestään taitorakenteiden suunnittelu ja mallintaminen tulisi toteuttaa algoritmisesti siitä saatavien hyötyjen takia (Särkkä 2015).

Ulkomailla algoritmiavusteista suunnittelua hyödynnetään hyvin monimuotoisissa projekteissa. Seuraavaksi esitellään case-kohde Kiinan Macausta. Kuvassa 16 on esitetty suunniteltava rakennus. Rakennus on 42-kerroksinen kaksitorninen hotelli. Se koostuu kahdesta betonirunkoisesta tornista ja alumiinisesta ulkopuolisesta rungosta. Betoni- ja alumiinirunko toimivat rakenteellisessa mielessä yhdessä. Kuvassa 17 on esitelty rakennuksen rungon eri osat. Alumiinirungossa on yli 2500 kappaletta haastavia liitoksia. Alumiinirungon haastavimmat osat löytyvät rakennuksen keskeltä, johon muodostuu kolme vapaamuotoista aukkoa. Haastavien muotojen ja rakennejärjestelmän takia sekä koko rakennuksesta että kaikista liitoksista tehtiin FEM-analyysit. Rhinoceros 3D ja Grasshopper -ohjelmistoja hyödynnettiin kaikkien näiden hallintaan. Niiden avulla saatiin tehtyä Autodeskin Robot Structural Analysis -ohjelmistoon FEM-analyysit liitoksista. MIDAS -ohjelmistolla tehtiin kokonaisstabiliteetin tarkastelu. Koska liitoskappaleita oli niin valtava määrä, niitä ei voitu tarkastella koko rakennuksen FEM-mallissa samaan aikaan, sillä yksi laskentakerta Midaksella kesti 12 tuntia. Tällöin jokainen muutos parametriseen malliin kestäisi 12 tuntia (Piermarini et al. 2016).

Kuva 16. *Macaun City of Dreams -hotelli (Piermarini et al. 2016, p. 57).*

Alumiinirungon algoritmiavusteinen suunnitteluprosessi jaettiin viiteen vaiheeseen: Tunnistetaan saman tyyppiset liitokset, liitosten detaljikaan suunnittelu arkkitehdin vaatimusten ja toteutettavuuden avulla, kokonaisstabiliteettitarkastelusta määrätään jokaisen liitoksen kuormat, suoritetaan FEM-laskenta ja lopulta tehdään raportointi ja valmistuskuvat. Grasshopperia hyödyntämällä luotiin algoritmi, joka pystyi geometrioiden ja liittyvien kappaleiden lukumäärien mukaan määrittämään saman tyyppiset liitostyypit. Kun liitostyypit oli löydetty, liitosten kokoonpanon eri osat suunniteltiin. Tässä reunaehtoja asetti vaativa arkkitehtuuri, koska kaikki liitokset olivat osa rakennuksen arkkitehtuuria ja visuaalista ilmettä. Kokoonpanojen toteutettavuus oli toinen tärkeä reunaehtoja asettava tekijä. Grasshopperin ja Rhinon avulla pystyttiin tehokkaasti havainnoimaan, jos tietyt liitoskomponentit eivät täyttäneet toteutettavuuden tai arkkitehdin asettamia vaatimuksia. Samalla tavalla liitosten voimasuureiden lisäämisessä ja laaduntarkastuksessa kyettiin hyödyntämään ohjelmistojen visuaalisuutta (Piermarini et al. 2016).



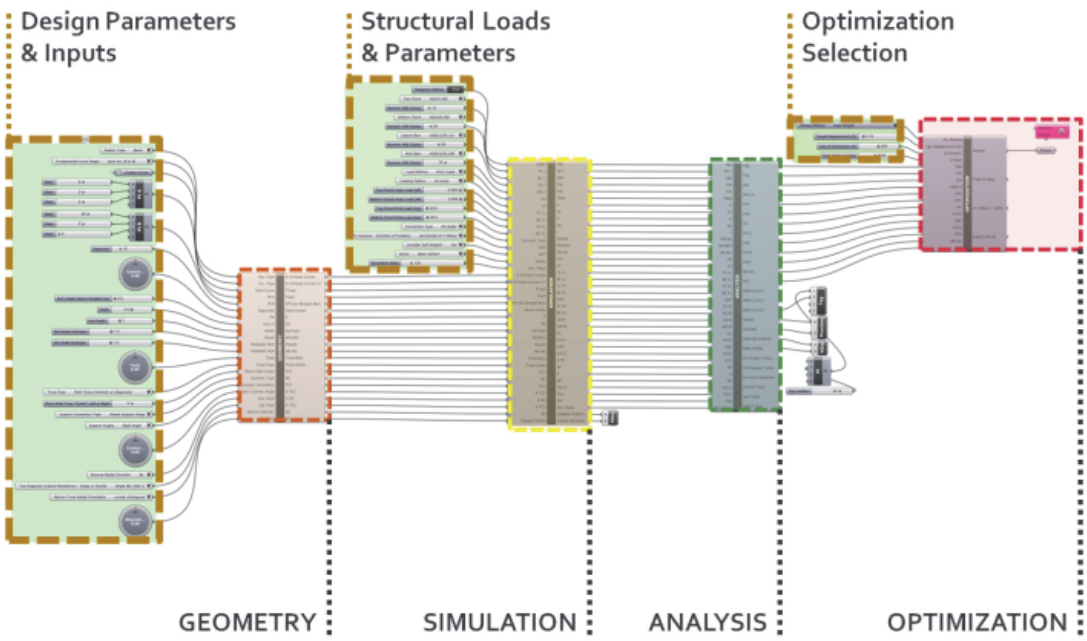
Kuva 17. *Kohteen rakennejärjestelmä koostuu kahdesta eri rungosta, jotka toimivat yhtenä kokonaisuutena (Piermarini et al. 2016, p. 58).*

Liitosten analysoinnissa hyödynnettiin FEM-laskentaa. Tämä työ pyrittiin automatisoimaan mahdollisimman pitkälle hyödyntäen eri ohjelmistojen rajapintoja siirtämään haluttua dataa ohjelmien välillä. Algoritmi loi ensiksi Rhinoon liitoksen pinnat, ne siirrettiin Robottiin, Midaksesta tuotiin liitokseen kuuluvat sauvat ja geometriat, sen jälkeen Robotissa luotiin FEM-verkosto ja toteutettiin analyysi. Kun tulokset analysoitiin, tarkastettiin ja hyväksyttiin, tuotettiin dokumentointi ja valmistuspiirustukset. Tähänkin

hyödynnettiin Grasshopperin tarjoamaa automatisointia. Jokaisesta kokoonpanosta tuotettiin automaattisesti dokumentointi, josta selviää laskentadokumentit ja valmistuskuvat. Projektin suunnittelijat toteavat, että alussa käytetty 4 kuukautta Grasshopperin koodin luomiseen mahdollisti tämän tasoisen suunnittelun. Alussa tehty hyvä koodi mahdollisti lopussa dokumentoinnin ja analysoinnin todella vähäisellä ajankäytöllä. Näillä uusilla työkaluilla suunnittelijat kykenevät hallitsemaan valtavia määriä tietoa, jota tavallisin keinoin ei ole mahdollista tai järkevää toteuttaa. Ajankäytön säästö toistuvassa yksinkertaisissa tehtävissä mahdollistaa insinöörien käyttävän aikaa oikeaan insinöörin suunnittelutyöhön (Piermarini et al. 2016). Algoritmiavusteista suunnittelua on sovellettu myös monissa muissakin viimeaikaisissa hankkeissa (Shepherd 2011; Lewis 2016).

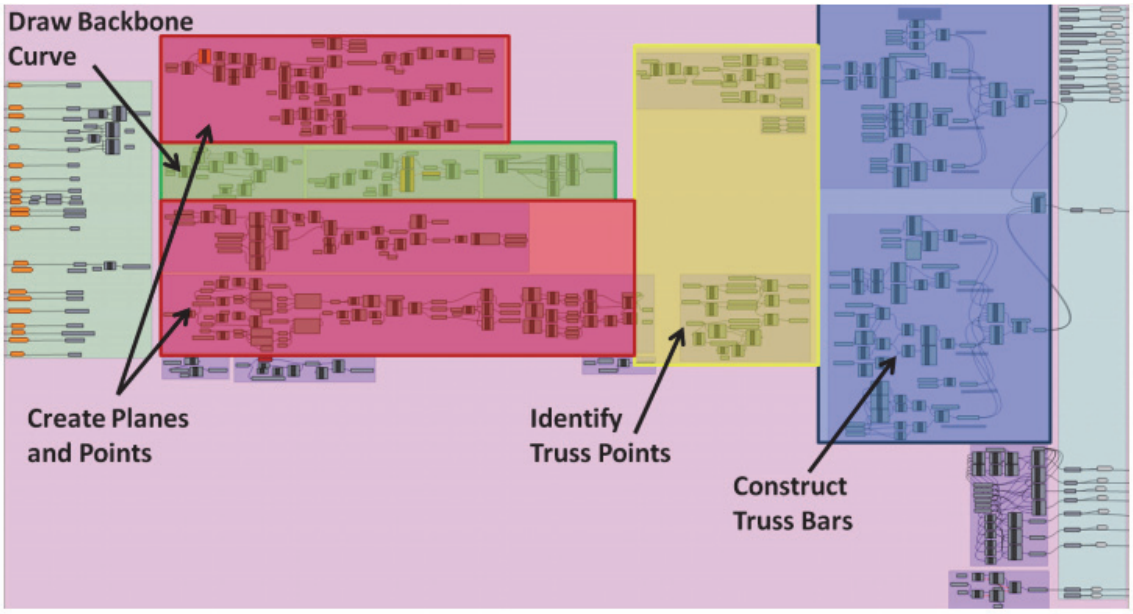
3.5.3 Rakenteiden optimointi Galapagoksella

Rakenteiden optimointi on monimutkainen prosessi. Makris (2013) luo työssään teräsrakenteisen ristikon optimointityökalun käyttäen Grasshopperia. Prosessi hyödyntää Grasshopperia, Rhinoa, Kangaroota ja Galapagosta. Grasshopper toimii koodin alustana, Rhino mallintamisen alustana, Kangaroo simuloinnin laskentaohjelmana ja Galapagos rakenteen optimointityökaluna. Monimutkaisen prosessin käsittelyä varten hän jakoi prosessin neljään isompaan kokonaisuuteen. Ensimmäinen osa geometry määrittää ristikon geometrian. Käyttäjä säätää parametreja ja syötteitä halutulla tavalla. Nämä eri syötteet ja parametrit toimivat myöhemmin myös optimoinnin lähtökohtina. Kuvassa 18 on esitetty Makriksen optimointityökalun suunnitteluprosessi (Makris 2013, p. 55-59).



Kuva 18. *Rakenteen algoritmiavusteinen suunnittelu visuaalisessa ohjelmoinnissa on jaettu neljään erotettavaan osaan (Makris 2013, p. 56).*

Geometrian luomisen osiolla pystytään määrittämään esimerkiksi ristikon tyyppi, uumasauvan profiili tai useampi profiili, ristikon korkeus ja tukiehdot. Osa näistä määrityksistä vaikuttaa olennaisesti prosessin muihin osa-alueisiin. Jokainen osa-alue kuvassa 18 sisältää enemmän koodia, mitä siinä esitetään. Makris on käyttänyt cluster-työkalua luodakseen käyttäjälle yksinkertaisemman käyttöliittymän. Tällä tavalla Makris voi ohjata käyttäjää muuttamaan vain haluttuja ja oikeita parametreja, jolloin työkalun toiminta on varmempaa. Kuvassa 19 geometria-komponentti on avattu. Siitä voidaan nähdä, miten komponentin logiikka toimii. Komponentti koostuu käytännössä neljästä eri osa-alueesta, joilla kaikilla on tietty tehtävä ristikon geometrian luomisessa. Yksi osa määrittää ristikon päälinjan, toinen luo linjalle tasot ja pisteet, kolmas määrittää tasojen ja pisteiden avulla paarre- ja uumasauvojen liitoskohdat ja neljäs luo ristikon kaikki sauvat. Lopputuloksena komponentilla on ristikosta sauvamalli (Makris 2013, p. 70-75)



Kuva 19. *Geometria-komponentin koodi avattuna (Makris 2013, p. 71).*

Seuraava osakokonaisuus on simulointikomponentti. Simulointikomponentin tehtävänä on muuttaa sauvamalli rakennemalliksi. Tämä komponentti koostuu viidestä eri osasta. Osat sisältävät jousimallien rakentamisen, kuormien määrittämisen, simuloinnin tekemisen ja sauvojen uudelleen mallintamisen. Simulointikomponentti on osakokonaisuuksista raskain laskentatehollisesti, joten sen koodin yksinkertaistamiseen Makris käytti eniten aikaa. Lopputuloksena komponentilla on uudelleen järjestetty lankamalli ristikosta (Makris 2013, p. 76-86).

Simulointikomponentista syötetään data analyysikomponenttiin. Simulointi toteutettiin kimmoteorian avulla, mutta analyysikomponentti tarkastelee lankamallia plastisuusteorian avulla. Käytännössä analysoinnissa tarkastellaan simuloinnin tuloksia plastisuusteorian avulla, eli selvitetään täyttääkö simuloinnin tulokset myös plastiset rajat. Tuloksista tehdään graafinen esitys, jossa sekä jännitykset että sallitut rajat ylittävät sauvat visualisoidaan. Tämän lisäksi ohjelma ilmoittaa tekstin muodossa mitkä sauvat ylittävät sallitut rajat (Makris 2013, p. 87-92).

Viimeiseksi kokonaisuuteen kuuluu optimointikomponentti. Optimointi toteutetaan Galapagos komponentilla, joka esiteltiin lyhyesti aikaisemmassa osiossa. Käyttäjän tulee valita tarkastelurajat ja komponenttiin muuttujat, joita optimoidaan. Tämän lisäksi on määritettävä, millä tavalla mahdolliset ratkaisut pisteytetään. Tässä tapauksessa muuttujaksi on valittu muun muassa ristikkotyyppi, sauvojen lukumäärä ja profiili, joita muuttamalla Galapagos luo, testaa ja pisteyttää eri vaihtoehdot (Makris 2013, p. 92-93).

Kuten mainittu, Galapagos tarvitsee menetelmän, jonka perusteella se pisteyttää eri ratkaisuvaihtoehdot. Tässä tapauksessa vaihtoehtoja oli viisi: kokonaispaino, kokonaispaino taipuma- ja siirtymärajoilla, sauvojen ja liitosten lukumäärä, hinta ja hinta taipuma- ja siirtymärajoilla. Näistä suunnittelija valitsee halutun tavan arvioida optimoinnin tu-

losta. Valinta ei kuitenkaan aina ole helppo tehdä, sillä esimerkiksi paino ja hinta ovat sidoksissa toisiinsa, samoin kuin sauvojen ja liitosten lukumäärä hintaan. Toisaalta sauvojen ja liitosten lukumäärä voi olla sidoksissa myös painoon ja paino vastaavasti on sidoksissa hintaan. Ei siis ole yksiselitteistä mikä näistä menetelmistä on paras arvioimaan ratkaisun laatua (Makris 2013, p. 94-98)

Rakenteiden optimointi on teoriassa iteratiivinen prosessi, huolimatta siitä, toteutetaanko se suunnittelijan vai ohjelmiston avulla. Ohjelmistolla toteutettu optimointi kykenee suunnittelijaa tehokkaammin etsimään mahdollisia vaihtoehtoja eri reunaehtojen avulla tarkasteltuna (Makris 2013, p. 157-160). Optimointia voitaisiin hyödyntää elementtisuunnittelussa esimerkiksi seinäelementtien elementtijakojen optimointiin. Siinä reunaehtoina voisi olla arkkitehtoniset seikat, nosturien asettamat elementtien painorajat, kuljetuksen aiheuttamat elementtien korkeus- ja painorajat ja liitosgeometria.

4. ELEMENTTISUUNNITTELU

Seuraavassa osiossa tutkimusta perehdytään elementtisuunnitteluun. Ensiksi tutkitaan lyhyesti elementtirakentamisen historiaa. Sen jälkeen tutkitaan elementtikauppamalleja ja niiden vaikutuksia elementtisuunnitteluprosessiin. Tämän tueksi esitellään Ramboll Finland Oy:n elementtisuunnitteluprosessin toteutusvaihe. Lopuksi luodaan väliseinäelementin algoritmiavusteinen suunnitteluprosessi ja tutkitaan algoritmiavusteisen suunnitteluprosessin kustannusvaikutuksia.

4.1 Elementtirakentaminen

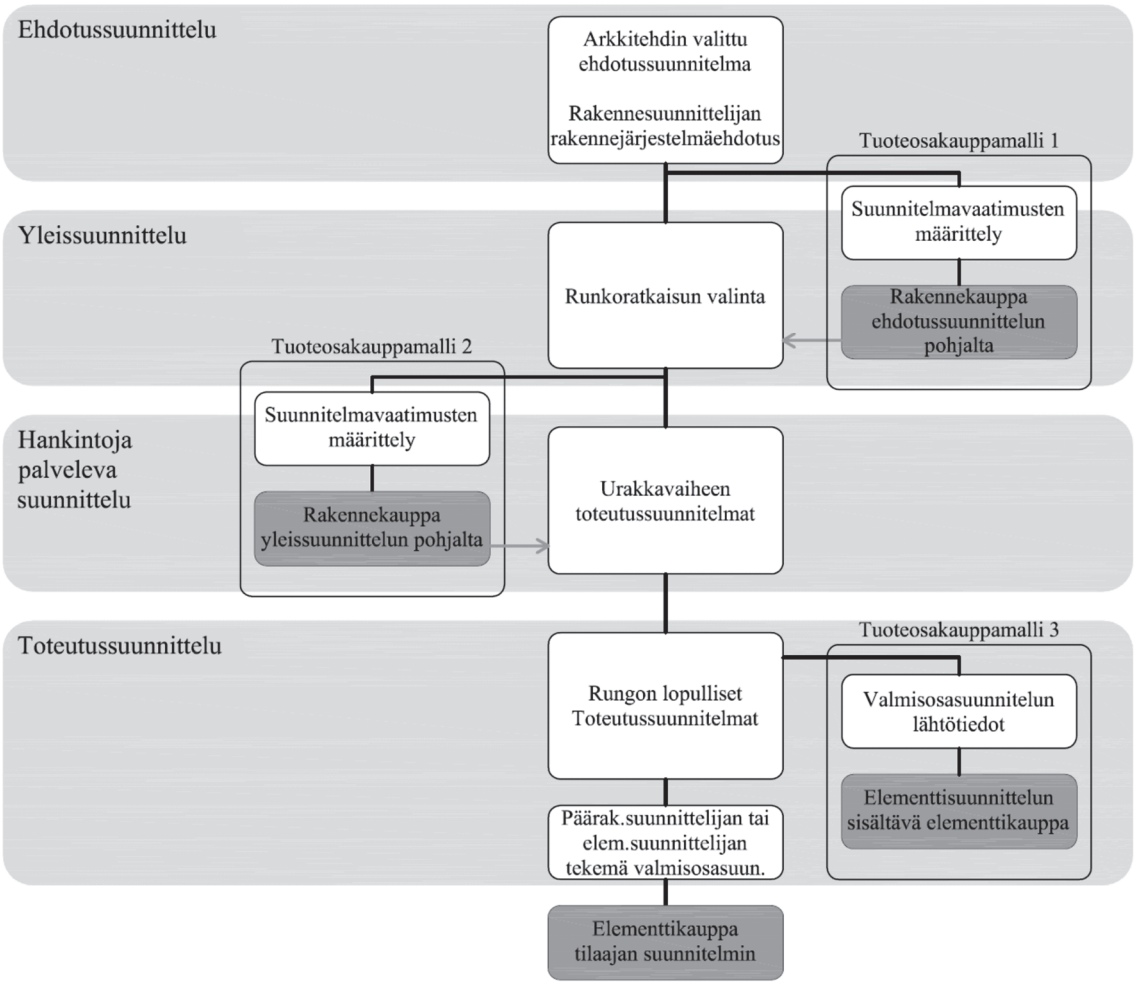
Betonielementti on muottiin valettava betonimassa, johon pääsääntöisesti lisätään raudoitteita ja muita valutarvikkeita. Betonielementtien valmistus Suomessa juontaa juurensa toisen maailmansodan ajoilta, 1940- ja 1950-luvuilta. Näihin aikoihin elementtitekniikka alkoi kehittyä Suomessa, mutta sen kehittymistä hidastutti sodan jälkeiset sotakorvaukset. Elementtitekniikan kehittyminen mahdollisti sotien jälkeiseen asuntopulaan vastaamisen. Uusien asuntojen rakentamiselle oli valtava kysyntä, joka vauhditti nopean rakentamisen kehittämistä (Elementtirakentamisen historia; Hytönen & Seppänen 2009).

Asuntorakentamisessa betonielementtien käyttö yleistyi 1950-luvulla. Tähän vaikutti vahvasti arkkitehtuurisuuntauksen sopiminen betonin ilmeelle. Alvar Aalto suunnitteli tähän aikaan betonisia tyyppitaloja, jotka vauhdittivat betonielementtien käyttöä. 1960- ja 1970-luvuilla elementtirakentaminen kiihtyi yhä. Tällöin maalta kaupunkiin muuttaminen oli vahvimmillaan, jolloin tarvittiin taas nopeasti ja taloudellisesti rakennettuja asuntoja. 1968-1970 vuosina kehitettiin elementtirakentamiseen betonielementtisysteemi eli BES-järjestelmä, jolla pystyttiin vastaamaan kiihtyvän elementtirakentamisen aiheuttamiin tarpeisiin ja niistä aiheutuviin ongelmiin. BES-järjestelmällä sovittiin elementtiteollisuudelle yhteisiä sääntöjä ja vakioituja mittajärjestelmiä, sekä esimerkiksi liitosdetaljeja. Elementtityypit haluttiin standardisoida, jonka avulla mahdollistettiin elementtien suunnittelu, valmistaminen ja asentaminen nopeasti ja kustannustehokkaasti (Elementtirakentamisen historia; Hytönen & Seppänen 2009).

4.2 Elementtikauppamallit

Elementtisuunnittelu voi olla mukana hankkeessa jo yleissuunnitteluvaiheessa riippuen valitusta elementtikauppamallista. Kuvassa 20 esitetään eri kauppamallit. Tavanomaisessa suunnittelu- ja rakentamisprosessissa elementtisuunnittelu toteutetaan pääosin

silloin, kun muut suunnitteluosapuolet ovat jo tehneet suunnittelutyönsä. Harmanen (2010, p. 51) kutsuu tätä kauppamallia elementtikauppa tilaajan suunnitelmin. Tämä kauppamalli vaatii pääsuunnittelijalta erityisen hyvän tietämyksen elementtitekniikasta ja -suunnittelusta. Pääsuunnittelijan vastuulle tässä mallissa jää muun muassa elementtien geometrian ja talotekniikan läpivientien vaatimien varausten suunnittelu. Tässä toteutusmallissa on myös huomattava, että siinä harvoin kilpailutetaan toteutusratkaisuita, joten rakennesuunnittelijalla tulee olla myös hyvä käsitys eri suunnitteluratkaisuiden kustannusvaikutuksista.



Kuva 20. *Elementtikauppamallien jaottelu hankkeen vaiheeseen sidottuna (Harmanen 2010, p. 49).*

Elementtikaupat voidaan toteuttaa myös tuoteosakauppoina. Tuoteosakauppamalli 1:ssä rakennuttaja ostaa kokonaisuuden, joka sisältää rakennuksen rungon ja/tai julkisivun suunnittelun ja usein myös asennuksen. Tällöin tarjoussuunnittelu tehdään arkkitehdin tilamallin pohjalta. Ehdotussuunnitelman laatu ja kattavuus on oltava riittävä, jotta runkoratkaisu voidaan yleissuunnitteluvaiheessa valita oikein perustein. Tämä malli soveltuu parhaiten rakennuksiin, joissa runko muodostaa suuren osan rakentamisen kustannuksista kuten teollisuushallit, liike- ja toimistorakennukset (Harmanen 2010, p. 53).

Tuotekauppamallia 2 voidaan käyttää yleissuunnitteluvaiheen jälkeen. Yleissuunnitteluvaiheessa runkoratkaisu on tehty ja kantavien rakenteiden sijainnit on päätetty, mutta dimensioissa voi vielä tapahtua muutoksia. Tämä antaa rakennetoimittajalle mahdollisuuden tarkentaa rakenneratkaisuja sopivammaksi omaan suunnitteluun ja tuotantoon. Malli 2 soveltuu usein parhaiten asuinrakennuksiin tai rakennuksiin, joissa on paljon kantavia betoniseiniä. Toimittaja pystyy vaikuttamaan elementtien kokoon ja saumoihin, tai esimerkiksi sisäkuorielementtien valmistukseen ja suunnitteluun (Harmanen 2010, p. 56).

Tuoteosakauppa 3 on elementtisuunnittelun sisältävä elementtikauppa. Rakennetoimittaja valitaan pää- ja TATE-urakoitsijoiden kanssa samaan aikaan hankintoja palvelevan suunnittelun jälkeen. Toteutussuunnitteluvaihetta ennen rakennesuunnittelija arvioi elementtijakoa ja rakenteiden paksuuksia erilaisten diagrammien avulla, jotta elementtisuunnittelijalla olisi mahdollisimman hyvät lähtökohdat toteuttaa lopullista suunnittelua. Yleisessä käytössä on myös sekamalli, jossa tilaajan valitsema elementtisuunnittelija vastaa elementtien geometrian, kuormien, vaatimusten ja tartuntaosien suunnittelusta, mutta punostuksen ja joskus myös raudoituksen suunnittelun toteuttaa eri tuoteosatoimittaja (Harmanen 2010, p. 59).

Tässä työssä keskitytään perinteiseen malliin, eli elementtikauppa tilaajan suunnitelmin. Tuoteosakaupat keskittyvät enemmän rakennesuunnittelijan näkökulmaan, joka ei palvele tämän tutkimuksen tavoitteita. Jos tutkittaisiin eri tuoteosakauppojen soveltuvuutta algoritmiavusteiseen suunnitteluprosessiin, työ olisi laajuudeltaan liian suuri. Tämä johtuu siitä, että eri tuoteosakaupat solmitaan hankkeen eri vaiheissa. Jokaisessa hankkeen vaiheissa tietomallien sisällön tarkkuus on eri. Lisäksi eri tuoteosakaupoissa tehtäväjako rakenne- ja elementtisuunnittelijan kesken on hyvinkin erilainen (Harmanen 2010). Lopputulemana kuitenkin kaikilla eri kauppamalleilla on toteutusvaiheen tietomalli, joka tehdään tavanomaisessa suunnitteluprosessissakin. Todetaan siis, että tutkimuksessa on järkevää tutkia ainoastaan perinteisen mallin elementtikaupan soveltuvuutta algoritmiavusteiseen suunnitteluun.

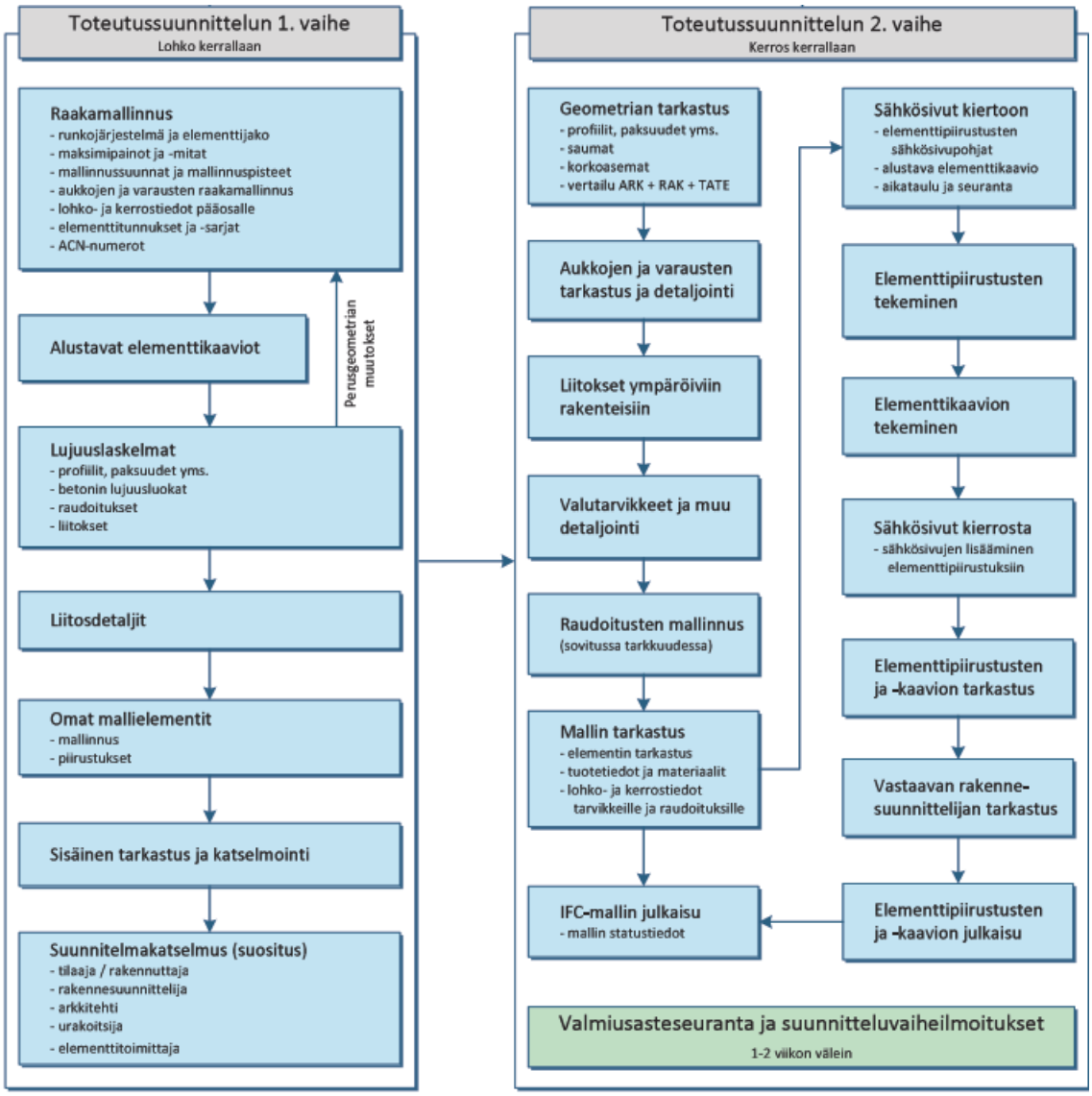
4.3 Elementtisuunnitteluprosessi

Edellisessä osiossa todettiin elementtikaupan mallin vaikuttavan oleellisesti elementtisuunnitteluun. Elementtikaupan malli vaikuttaa siihen, kuka elementtisuunnittelun tekee, missä vaiheessa se aloitetaan, onko suunnittelu paloiteltu vielä pienempiin osiin. Lisäksi tehtävien vastuualueet voivat muuttua. Tässä työssä on tavoitteiden mukaisesti tarkoitus tutkia miten algoritmiavusteinen suunnittelu vaikuttaa elementtisuunnitteluprosessiin. Elementtisuunnitteluprosessi kulminoituu toteutussuunnitteluvaiheeseen, jossa tietomallintaminen on tärkeässä roolissa. Näistä syistä tämä osio käsittelee ainoastaan toteutussuunnitteluvaiheen konkreettisesti mallintamiseen vaikuttavia asioita.

Päärakennesuunnittelija	Elementtisuunnittelija
• Käytettävä mitoitusnormisto • Kokonaisstabiliteettilaskelmat ja jäykistysvoimia välittävät liitokset. • Rungon työnaikainen kokonais-vakavuus • Kuormitustiedot ja vaatimukset • Reikätietietojen antaminen ja reikien sijoittelun koordinointi • Paikallavalurakenteet • Tyyppielementit • Rakennusfysikaalinen suunnittelu • Tyyppiliitokset • Koordinoi ja yhteensovittaa eri valmisosasuunnittelijoiden työtä • Riittävä elementtien rakenteellinen tarkastus • Viranomaishyväksyntä • Asennussuunnitelman tarkastus ja hyväksyntä • Suunnitteluratkaisujen työturvallisuudesta huolehtiminen • Rakennuksen käyttö- ja huolto-ohje rakenteiden osalta • Rakenteellisen turvallisuuden riskien arviointi	• Lähtötietojen yhteensopivuuden varmistaminen • Elementtien lujuuslaskelmat (murto- ja käyttörajatila, onnettomuusrajatila, palotila) • Jäykistysvoimia välittämättömät liitokset. • Kaikki elementtien valmistus-suunnitelmat • Elementtien liitos- ja asennusdetaljit • Yksittäisten elementtien asennusai-kainen vakavuus ja tuentasuunnitelmat • Turvalaitteiden vaatimat tartunnat • Elementtikaaviot • Elementti- ja valutarvikeluettelot • Elementtien vaatimat tartunta-suunnitelmat • Asennussuunnitelman tarkastus ja hyväksyntä tarvittaessa

Kuva 21. *Päärakenne- ja elementtisuunnittelun välinen työnjako (Harmanen 2010, p. 72).*

Perehdytään elementtisuunnitteluprosessiin kuvan 21 avulla. Harmanen (2010, p. 72) esittää kuvassa 21 päärakenne- ja elementtisuunnittelijan tehtävien jaon. Keskitytään elementtisuunnittelijan osuuteen. Elementtisuunnittelijan tulee varmistaa lähtötietojen yhteensopivuus, joka käytännössä tarkoittaa, että lähtötiedot ovat sovitun mukaiset ja ne ovat käytettävissä sovituilla ohjelmistoilla. Elementtisuunnittelija on vastuussa jokaisesta elementistä, joka lähetetään elementtitehtaalle valmistettavaksi, joten elementtien lujuuslaskenta on toteutettava. Elementtien ja paikallavalettavien rakenteiden väliset liitokset on suunniteltava ja osoitettava niiden kestävän jäykistävät voimat. Valmistettavista elementeistä on tehtävä sopimuksen mukaiset valmistuskuvat. Suunnittelua ja asennusta varten elementtien liitos- ja asennusdetaljit on suunniteltava. Työnaikaista tilannetta varten turvalaitteiden tartunnat, elementtien asennus ja työnaikainen tuenta on suunniteltava. Tuotantoa ja työmaan asennusta helpottamiseksi on tuotettava elementtikaavioita, elementti- ja valutarvikeluetteloita.



Kuva 22. *Ramboll Finland Oy:n elementtisuunnitteluprosessin toteutussuunnitteluvaihe (Elementtisuunnittelun prosessikaavio, 2017).*

Elementtisuunnittelijan tehtävistä muodostuu lähtökohta elementtisuunnitteluprosessille. Tässä osassa esitettävä mallintava elementtisuunnitteluprosessi perustuu rajattuun osaan Ramboll Finland Oy:n elementtisuunnittelun prosessikaaviosta (Elementtisuunnittelun prosessikaavio, 2017). Kaaviosta käsitellään toteutussuunnitteluvaiheeseen liittyvä osuus, joka on esitetty kuvassa 22.

4.3.1 Toteutussuunnittelun 1. vaihe

Toteutussuunnittelun valmisteluosassa pyydetään lähtötiedot elementtisuunnittelua varten. Tilaaja, arkkitehti, rakennesuunnittelija, LVI-suunnittelija, sähkösuunnittelija, urakoitsija ja elementtitoimittajat toimittavat elementtisuunnittelijan pyynnön mukaisesti omat lähtötietonsa. Lähtötiedot tarkastetaan ja annetaan täydennys- ja tarkennuspyyntöjä, jos havaitaan puutteita. Toteutussuunnitteluun kuuluu yhtenä osana projektinhallinta,

mutta sitä ei käsitellä tämän tutkimuksen yhteydessä. Toteutussuunnittelu jaetaan kahteen vaiheeseen. Ensimmäisessä vaiheessa lähtötietojen pohjalta ryhdytään luomaan raakamallia. Raakamalli sisältää runkojärjestelmän, aukkojen ja varausten mallintamisen. Ensimmäisen vaiheen suunnittelu tehdään lohko kerrallaan. On tärkeää, että mallintaminen tehdään jo alusta alkaen sovituilla tavoilla toteuttaen hyviä mallinnuskäytäntöjä. Näihin kuuluu muun muassa mallinnussuunnan, -pisteiden, lohko-, kerrostietojen, elementtitunnusten ja juoksevien numeroiden hallintaa. On hyvin tärkeää, että mallintavat osapuolet ovat sopineet näistä käytännöistä jo ennen kuin mallintamista aloitetaan.

Raakamallista tehdään alustavat elementtikaaviot ja -luettelot. Elementtitoimittaja voi näiden avulla suunnitella omaa tuotantoaan. Lisäksi ne auttavat elementtisuunnittelijaa tarkentamaan omaa resursointia ja hahmottamaan suunnittelutyön lopullista laajuutta ja haastavuutta. Alustavien elementtikaavioiden ja -luetteloiden jälkeen suoritetaan elementtien lujuuslaskenta. Lujuuslaskennan perusteella elementtien dimensioita ja materiaaliarvoja voidaan muuttaa. Mahdollisten muutosten jälkeen suunnitellaan elementtien ja paikallavalurakenteiden väliset liitosdetaljit. Tämän vaiheen detaljisuunnittelu tehdään CAD-pohjaisesti, mutta suunnittelun lähtökohtana ja apuna käytetään tietomallia. Yleisesti ottaen liitosdetaljit ovat hankalin ja eniten työtä vaativa osuus. Detaljeihin tulee usein muutoksia myöhemmissä vaiheissa, joita ei välttämättä tässä vaiheessa ole osattu ottaa huomioon.

Oman laadunvarmistuksen ja -tarkastuksen tueksi detaljoinnin jälkeen tehdään jokaisesta elementtityypistä mallielementit. Mallielementeiksi valitaan yleensä vaikeimpia elementtejä, jotka sisältävät mahdollisimman paljon erilaisia toistuvia yksityiskohtia. Esimerkiksi ulkoseinissä voi olla kolmea erilaisesti toistuvaa elementtityyppiä. Tällöin on järkevintä tehdä kaikista kolmesta elementtityypistä mallielementit. Kaikista mallielementeistä tehdään myös mallipiirustukset. Mallielementtejä verrataan muihin samanlaisiin elementteihin, jolloin mahdolliset puutteet tai selkeät virheet ovat yleensä helposti havaittavissa. Mallielementtien mallipiirustuksia käytetään myös piirustusten kloonaamiseen. Tämä mahdollistaa yhä tehokkaamman piirustustuotannon.

Kaikki edellä mainitut tuotokset on tarkastettava. Tässä vaiheessa malliin voidaan tehdä vielä kohtuullisen pienellä työmäärällä muutoksia. Toteutussuunnittelun ensimmäisen vaiheen lopuksi on suositeltavaa toteuttaa suunnitelmakatselmus kaikkien projektin osapuolten kesken. Tällä varmistetaan, että elementtisuunnittelija on saanut kaikki lähtötiedot ja ymmärtänyt niiden sisällön. Katselmuksella pystytään tehokkaasti tarkastamaan kaikkien projektin osapuolten saavan sovituilla tavoilla suunnitellut elementit. Tilaajalle ja elementtitoimittajalle tämä on varsinkin hyvä tilaisuus tarkastaa suunnitelmien vastaavan sovittua.

4.3.2 Toteutussuunnittelun 2. vaihe

Toteutussuunnittelun toisessa vaiheessa detaljoidaan elementit. Tässä vaiheessa suunnittelu ja mallintaminen tehdään kerros kerrallaan. Ennen detaljointia tarkastetaan sekä rakennuksen että rakennusosien geometria. Mallista tarkastetaan seinien, välipohjien ja muiden rakennusosien sijainti ja korkoasema. Rakennusosien profiilit ja paksuudet tarkastetaan myös. Tarkastus tehdään vertaamalla tietomallia arkkitehdin, rakennesuunnittelijan ja talotekniikkasuunnittelijoiden IFC-malleihin tai muuhun lähtötietoaineistoon. Tämän jälkeen tehdään varausten ja aukkojen tarkastus.

Toteutussuunnitteluvaiheen kriittisiksi vaiheiksi voidaan todeta geometrian, aukkojen ja varausten tarkastus. Näillä koetaan olevan hyvin suuria vaikutuksia suunnitteluprojektin onnistumisen kannalta. Tämä perustuu siihen, että mitä aikaisemmassa vaiheessa mallintamista tai kuvatuotantoa näihin liittyvät virheet tai ristiriitaisuudet todetaan sitä helpompi ja vähätöisempi ne ovat korjata. Tämän lisäksi, jos huomaamatta jää esimerkiksi kantaviin rakenteisiin liittyvä muutos arkkitehdin tai talotekniikan suunnitteluissa, on sillä kustannusmielessä yleensä hyvin suuri vaikutus. Tarkastuksilla pyritään varmistamaan, että suunniteltavat elementit vastaavat kaikkien osapuolten yhteistä ymmärrystä, sekä pyritään välttämään elementtikuvien revisioita tai pahimmassa tilanteessa reklamaatioita. Yleensä tässä vaiheessa on tullut muutoksia alustaviin varauksiin tai aukkoihin, joten niiden tarkastaminen ja tekeminen malliin ovat äärimmäisen tärkeitä vaiheita. Oletettavasti muutoksia tulee myös myöhemmissä vaiheissa projektia, jolloin muutostenhallinta on kriittisessä roolissa.

Tarkastusten jälkeen alkaa elementtien detaljointi. Sekä ohjelmistoteknisistä että kokemusperäisistä syistä on elementtien detaljointiin syntynyt tietty järjestys. Ensimmäisenä mallinnetaan elementteihin liitokset niitä ympäröiviin rakenteisiin. Nämä mallinnetaan suunniteltujen detaljien perusteella. Kun liitokset on mallinnettu, mallinnetaan tarvittavat valutarvikkeet ja muu detaljointi. Valutarvikkeisiin luetaan muun muassa nostotarvikkeet, kappaleet tuentaa varten ja turvalaitteet. Muuta detaljointia ovat muun muassa erilaisten kiinnityslevyjen suunnittelu kevyitä rakenteita varten, valesaumat ja elementtien kuljetustuet. Teklasta johtuvista syistä raudoitteiden mallinnus on suositeltavaa tehdä viimeisenä. Raudoitustyökalut ovat kohtalaisen herkkiä rikkoutumaan, jos raudoitteet on mallinnettu elementtiin, johon tehdään muutoksia.

Elementtien sähkötarvikkeiden merkintään on olemassa useampi tapa, mutta tässä esitellään kaksi yleisintä Ramboll Finland Oy:ssä käytössä olevaa tapaa. Ensimmäisenä vaihtoehtona on, että sähkösuunnittelija mallintaa sähkötarvikkeet suoraan elementteihin. Tämä on yleistyvä tapa. Tässä vaihtoehdossa sähkösuunnittelija mallintaa sähkötarvikkeita elementteihin, joihin elementtisuunnittelija ei enää tee muutoksia. Elementtisuunnittelija kuitenkin mitoittaa sähkötarvikkeet elementtikuvaan, kun sähkösuunnittelija on mallintanut sähkötarvikkeet ja tarkastanut niiden oikeellisuuden.

Toinen vaihtoehto on toteuttaa raudoitteiden mallintamisen jälkeen elementin ja sen kaikkien osien, tuotetietojen, materiaalien, lohko- ja kerrostietojen tarkastus. Sähkötarvikkeita ei tässä vaihtoehdossa lisätä malliin, vaan ne lisätään elementin valmistuspiirustukseen omalle sivulleen. Käytännössä on kaksi vaihtoehtoa, kuinka sähkösuunnittelija lisää sähkötarvikkeet kuvaan. Sähkötarvikkeet voidaan tuoda elementtikuvaan ensiksi luomalla tietystä seinälinjasta alustava elementtikaavio, johon sähkösuunnittelija lisää sähkötarvikkeet elementtikohtaisesti. Kaaviosta jokaisen elementin sähkötarvikkeet voidaan siirtää seinälinjan elementtien valmistuskuviin. Toinen vaihtoehto on luoda elementeistä alustavat elementtikuvat. Alustavilla kuvilla näissä kahdessa tapauksessa tarkoitetaan sitä, ettei kuviin vielä mitoiteta kuin sillä hetkellä tarpeelliset mittatiedot. Sähkösuunnittelijalle lähetetään jokaisen elementin sähköistyskuva DWG-muodossa, jossa esitetään elementin geometria, aukot ja muut asiat jotka voivat vaikuttaa elementtien sähkösuunnitteluun. Sähkösuunnittelija lisää sähkötarvikkeet DWG-kuvaan, jonka jälkeen sähkösuunnittelija lähettää valmiit sähköistyskuvat takaisin elementtisuunnittelijalle. Elementtisuunnittelija tuo DWG-kuvan takaisin kyseiseen elementtikuvaan.

Sähkötarvikkeiden suunnittelutavasta riippuen elementtikuvien tekemisen ajankohta vaihtelee jonkin verran. Ensimmäisessä mallintavassa suunnittelutavassa elementtikuvat voidaan riippuen sähkösuunnittelijan tilanteesta mallintaa joko ennen sähkötarvikkeiden mallintamista tai niiden jälkeen. Toisessa vaihtoehdossa alustavat elementtikuvat luodaan heti kun elementtisuunnittelija on tarkastanut elementtien oikeellisuuden tai niitä aletaan tehdä, kun alustavat elementtikaaviot on lähetetty sähkösuunnittelijalle sähköistettäväksi. Kun sähkösuunnittelijalta saadaan sähköistyskuvat takaisin, ne lisätään kyseisten elementtien valmistuskuviin. Näiden elementtien kuvat tehdään sähköjen lisäämisen jälkeen loppuun. Tietyn kerroksen tai rajatun alueen elementtikuvien valmistuttua elementtikaaviot ja -luettelot tehdään loppuun.

Kun elementtikuvat, -kaaviot ja -luettelot ovat valmiita, elementtisuunnittelija tarkastaa ne kaikki vielä. Elementtisuunnittelijan tarkastuksen jälkeen myös vastaava rakennesuunnittelija tarkastaa ne. Kun tarkastukset on tehty hyväksytysti, elementtikuvat, -kaaviot ja -luettelot julkaistaan. Riippuen sovituista toimintatavoista, ne lähetetään elementtitoimittajalle ja julkaistaan projektipankissa. Näiden lisäksi tietomallista tehdään IFC-malli, joka julkaistaan myös sovituin väliajoin projektipankissa. Tärkeänä seikkana tästä esitetystä prosessista on huomata, että useat näistä vaiheista toteutetaan limitetysti. Edellä esitetty prosessi kuvastaa kuitenkin esimerkiksi yhden tietyn kerroksen yhden elementtityypin suunnitteluprosessia.

4.4 Väliseinän algoritmiavusteinen suunnitteluprosessi

Tässä osassa tutkitaan väliseinän algoritmiavusteista suunnitteluprosessia. Prosessista luodaan havainnollistamista varten prosessikaavio. Diplomityön aikaisempien osioiden avulla ongelmaa on lähdetty purkamaan osiin. Pohjana on hyödynnetty olemassa olevaa tietomallintavaa elementtisuunnitteluprosessia ja kirjallisuudessa esitettyjä hajauta ja

hallitse -tekniikoita ja muita ryhmittelyyn ja jäsentelyyn liittyviä tekniikoita. Prosessista tehtiin Business Process Model and Notation (BPMN 2.0) -standardin mukaisesti prosessikaavio, joka on esitetty liitteessä A.

Prosessi on jaettu neljään eri osaan: lähtötietojen kokoaminen ja tarkastus, väliseinien erottelu lähtötietoaineistosta, väliseinien päägeometrialinjojen luominen, muokkaus ja väliseinien algoritmiohjattu mallintaminen ja väliseinien algoritmiohjattu detaljointi. Prosessissa on tehty tiettyjä yksinkertaistuksia. Lähtötilanne elementtisuunnittelussa on, että elementtisuunnittelu toteutetaan täysin arkkitehdin IFC-mallin pohjalta. Mahdolliset muut lähtötilanteet jätetään tutkimatta tässä työssä. Tietomallinnusohjelmistona käytetään Teklaa ja algoritmin luomiseen Rhinoa ja Grasshopperia. Prosessi ei kuitenkaan ole sidonnainen näihin ohjelmistoihin, mutta havainnollistamisen vuoksi niitä käytetään esimerkkeinä prosessissa.

4.4.1 Lähtötietojen kokoaminen ja tarkastus

Algoritmiavusteinen suunnitteluprosessi alkaa lähtötietoaineiston kokoamisesta. Kootaan kaikki lähtötiedot, jotta voidaan siirtyä algoritmiavusteiseen suunnitteluun. Lujuuslaskenta on tässä vaiheessa tärkeässä roolissa, sillä sen avulla saadaan tarvittavat materiaalitiedot, profiilit ja raudoitteet selville. Tässä työssä ei kuitenkaan sen tarkemmin perehdytä lujuuslaskentaan vaan tehdään oletus, että lujuuslaskennasta saadaan vaaditut lähtötiedot suunnittelua varten.

Kaikelle lähtötiedolle on asetettava algoritmiseen suunnitteluprosessiin soveltuvat lähtötietovaatimukset, jotta suunnittelu voidaan toteuttaa prosessikaavion mukaisesti. Algoritmit ovat hyvin tarkkoja niihin syötettävästä tiedosta. Näistä syistä projektikohtaisesti on sovittava eri osapuolten kesken mitä ja missä muodossa lähtötiedot ovat. Kun lähtötiedot on saatu ja koottu yhteen on tarkastettava, että ne ovat asetettujen vaatimusten mukaiset. Jos ne eivät ole, ne palautetaan lähtötiedon toimittaneelle osapuolelle korjattavaksi.

4.4.2 Väliseinien erottelu IFC-mallista

Arkkitehdilta saadaan rakennuksen IFC-malli, josta on tarkoitus erotella väliseinät. Arkkitehdit käyttävät useita eri ohjelmistoja mallintamiseen, mikä aiheuttaa vaihtelevuutta lähtötiedoksi saatavien IFC-mallien tietosisällölle. Tästä syystä on olennaista asettaa vaatimukset arkkitehtien malleille ja tarkastaa niiden paikkansapitävyys. Arkkitehtien on asetettava väliseinille riittävästi ominaisuuksia, jotta kantavat väliseinät kyetään erottelemaan muista rakennusosista.

Erotteluun voidaan käyttää muun muassa SimpleBim-ohjelmistoa. Sen avulla päästään tutkimaan ja uudelleen järjestelemään IFC-mallin tietoa. Väliseinien erottelussa on tärkeää, että arkkitehti on noudattanut sovittuja vaatimuksia. Tällöin voidaan luottaa sii-

hen, että filtteröintiin käytettävät ominaisuudet toimivat odotetulla tavalla. Erotellut seinät tulee tarkastaa, kun erottelu on toteutettu. Jos huomataan että arkkitehdin IFC-malli tai erotellut seinät eivät vastaa näitä vaatimuksia, on vaihtoehtona palauttaa malli arkkitehdille korjattavaksi tai tehdä itse korjaukset SimpleBimillä. Molemmissa tapauksessa tehdään ilmoitus arkkitehdille, jotta he korjaavat asian omaan malliinsa. Lopputuloksena tällä osiolla on väliseinät erillisessä IFC-tiedostossa.

4.4.3 Väliseinien päägeometrialinjojen luominen, muokkaus ja algoritmiohjattu tietomallintaminen

Arkkitehdin IFC-mallista eroteltiin halutut väliseinät omaan IFC-tiedostoon. IFC-tiedoston siirtoon tarvitaan erillinen ohjelma, jotta se saadaan siirrettyä algoritmiavusteisen suunnittelun ohjelmistoon. Tähän voidaan käyttää esimerkiksi Geometry Gymiä, jolla IFC-tiedosto saadaan siirrettyä Rhinoon.

Seuraavaksi tarkoituksena on luoda päägeometria IFC-mallin seinien perusteella. Luodaan algoritmi, jossa samanaikaisesti luodaan seinien päägeometrialinjat, seinien korkeus, korot ja päägeometrialinjat jaetaan elementtijaon mukaisesti. Jokainen näistä osista sisältää useamman prosessin, joiden lopputuloksena saadaan korjattu päägeometria jaettuna elementtijaon mukaisesti. Päägeometria tarkastetaan Rhinossa arkkitehdin IFC-mallin avulla. Tarkastuksessa etsitään muun muassa oviaukkojen törmäyksiä elementtien välisiin liitoskohtiin.

Kun väliseinien päägeometrialinjat on tarkastettu ja hyväksytty, luodaan algoritmi, jolla väliseinäelementit mallinnetaan. Grasshopperiin luodaan algoritmi, jolla mallinnetaan väliseinät Teklaan. Edellisestä osasta saadaan lähes kaikki tarvittavat lähtötiedot, paitsi elementtien numerointiin, attribuutteihin materiaaleihin ja muihin vastaaviin liittyvät tiedot. Nämä tuodaan osaksi algoritmia, jolloin lopputulokseksi Teklaan saadaan mallinnettua väliseinäelementit, jotka sisältävät kaikki vaadittavat tiedot.

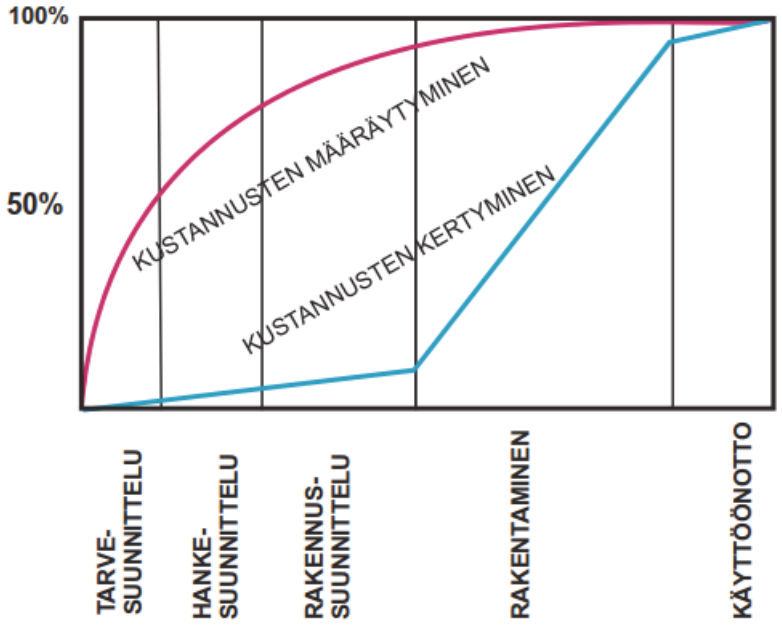
4.4.4 Väliseinien algoritmiohjattu detaljointi

Väliseinäelementtien detaljointiin kuuluu useita eri vaiheita, jossa tulee myös ottaa huomioon muut suunnitteluosapuolet. Detaljoinnin prosessi on hyvin samanlainen kuin perinteisessä mallintavassa suunnitteluprosessissa. Lähtötietona on edellisestä osasta saatavat väliseinäelementit. Ensimmäisenä detaljoinnissa väliseiniin tulee lisätä varaukset ja aukotukset. Arkkitehdin IFC-mallista tuodaan väliseinien oviaukot Rhinoon. Rhinossa tarkastetaan, että tuotiin ainoastaan aukot, ja että aukot ja niiden asemointi vaikuttavat oikeilta. Samalla tavalla tehdään LVIAS-suunnittelijoilta saatavat varaus- ja aukkotiedot. Tämän jälkeen luodaan algoritmi, joka luo Teklaan aukko- ja varauskappaleet ja rei’ittää seinät.

Näiden vaiheiden jälkeen tehdään detaljointi loppuun. Mallinnetaan liitokset ympäröiviin rakenteisiin, valutarvikkeet, muu detaljointi ja raudoitus. Jokaiselle näistä osista tulee tehdä oma algoritminsa. Kun detaljointi on tehty, Teklassa tarkastetaan kaikki väliseinät, joihin detaljointi on tehty. Jos virheitä tai puutteita havaitaan, palataan algoritmiin ja tehdään tarvittavat korjaukset, kunnes elementtien detaljointi hyväksytään. Tarkastus on hyvin tärkeä tehdä myös tässä vaiheessa, koska virheiden tai puutteiden korjaus tässä vaiheessa prosessia on vielä kohtalaisen helppoa. Jos virheitä tai puutteita havaitaan elementtien kuvatuotannon aikana, on virheiden korjaaminen hitaampaa ja työläämpää. Kun seinät on detaljoitu, tarkastettu ja hyväksytty, tehdään niistä valmistuskuvat.

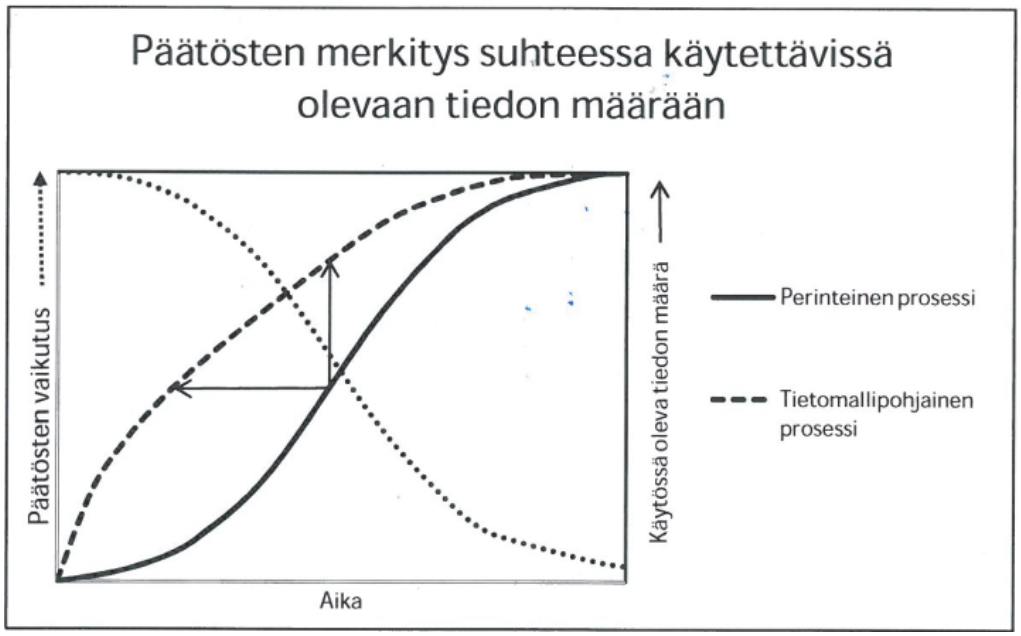
4.5 Algoritmiavusteisen suunnitteluprosessin kustannusvaikutukset

Rakennushankkeiden kustannusten hallinta on tärkeä osa projektinhallintaa. Ennen mallintavaa suunnittelua suunnittelun painopiste oli enemmän toteutussuunnitteluvaiheessa. Mallintavaan suunnitteluun siirryttäessä suunnittelun painopiste siirtyi enemmän luonnosvaiheen suunnitteluun. Painopisteen siirtyminen on perusteltavissa alla olevan kuvan 23 avulla. Mitä aikaisempaan vaiheeseen suunnittelua saadaan lisättyä, sitä enemmän kyetään vaikuttamaan suunnitteluratkaisuiden kustannuksiin. Eli mitä aikaisemmassa vaiheessa suunnittelua kyetään sitomaan kustannuksia perustuen suunnitteluun sitä vähemmän myöhemmin tulisi ilmetä suuria kustannuksiin vaikuttavia muutoksia (Pro IT -tutkimus, p. 3-5; Harmanen 2010, p. 28-30).



Kuva 23. *Kustannusten määräytyminen ja kertyminen eri hankevaiheisiin suhteutettuna (Pro IT -tutkimus, p. 4).*

Harmanen (2010, p. 28-29) käyttää tutkimuksessaan samaa kuvaa 23. Hän esittää, että kustannusten aikainen määräytyminen on ongelma tilaajan näkökulmasta, sillä perinteisessä suunnittelussa kustannusten määräytymiseen ei ole olemassa riittävästi tietoa tukemaan päätöksiä. Tilaajan edun mukainen tilanne olisikin, että kustannusten määräytyminen olisi yhä aikaisemmassa vaiheessa, mutta samassa suhteessa tiedon määrän tulisi kasvaa myös aikaisemmin. Tällöin suunnittelijoilla olisi käytössään mahdollisimman paljon tietoa suunnittelupäätösten tekemiseen jo varhaisessa vaiheessa. Kun suunnittelupäätösten kustannusvaikutuksista ymmärretään varhaisessa vaiheessa hanketta, tilaajan on helpompi sitoutua näihin kustannuksiin.



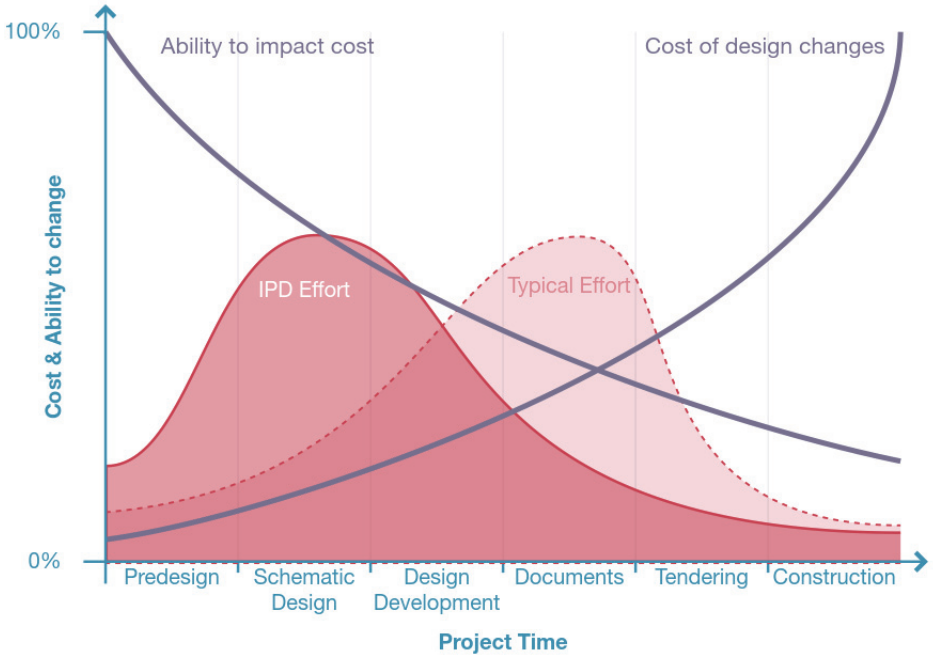
Kuva 24. *Perinteisen ja mallintavan prosessin vertailu: käytettävissä olevan tiedon määrä suhteutettuna päätösten vaikutuksiin (Harmanen 2010, p. 29).*

Harmanen esittää kuvassa 24 perinteisen ja tietomallintavan prosessin vertailun. Kuvaajassa esitetään, että tietomallintavalla prosessilla käyrä sijoittuu aikaisempaan vaiheeseen projektia verrattuna perinteiseen prosessiin. Tämä tarkoittaa sitä, että mallintavalla prosessilla kyetään tuottamaan enemmän ja nopeammin käytössä olevaa tietoa verrattuna perinteiseen prosessiin. Kuvaajan mukaan tämä mahdollistaa myös aikaisemmassa vaiheessa projektia merkittävämpien päätösten tekemisen. Tällöin kustannusten määräytyminen tapahtuu aikaisemmassa vaiheessa hanketta, mutta niiden määräytyminen ja lukitseminen perustuvat tosiasialliseen tietoon (Harmanen 2010, p. 28-30).

Kustannusten määräytyminen on yksi Davisin (2013) tohtorityön teema. Hänellä on kaksi näkökulmaa kustannuksiin: niiden määräytymisen ajankohta ja muutosten aiheuttama kustannusvaikutus. Kuvassa 25 hän esittää Patrick MacLeamyn luoman kuvaajan projektivaiheen suhteesta suunnitelmamuutosten kustannuksiin ja mahdollisuuteen vaikuttaa kustannuksiin. Tämä kuvaaja kuvastaa osittain samaa ideaa mitä Pro-IT tutkimuksen kuva 23 ja Harmasen kuva 24. Perinteisellä suunnitteluprosessilla ollaan typical effort alueella, jossa mahdollisuus vaikuttaa kustannuksiin on huono ja muutoksista johtuvat kustannukset ovat korkeat, koska suunnittelu tehdään hyvin myöhäisessä vaiheessa hanketta (Harmanen 2010, p. 28-30; Davis 2013, p. 32-36).

IPD effort tarkoittaa integrated project delivery, jonka ideana on ohjata kaikkia suunnitteluosapuolia siirtämään suunnittelun painoa varhaisempaan vaiheeseen projektia. Yhtenä tärkeänä osana IPD prosessia on tietomallintaminen. Mitä pitemmälle hankkeessa mennään, sitä enemmän isoja päätöksiä, joilla on myös suuri kustannusvaikutus, on ollut pakko lukita jo aikaisessa vaiheessa. Perinteisellä prosessilla näiden päätösten taustalla ei ole ollut yleensä riittävää tietoa. IPD prosessilla pyritään ohjaamaan suunnittelua

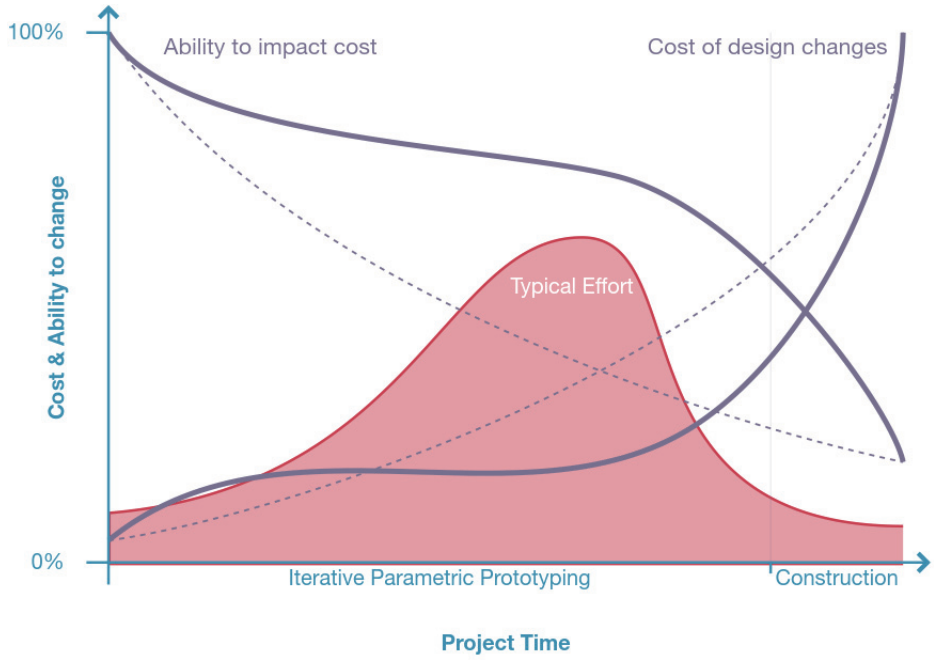
nimenomaan tästä syystä aikaisempaan vaiheeseen, jotta isojen kustannusten kiinnittämiseen on riittävästi tietoa (Harmanen 2010, p. 28-30; Davis 2013, p. 32-36).



Kuva 25. *MacLeamy-kuvaaja kustannuksiin vaikuttamisesta ja muutosten kustannusten suuruudesta perinteisellä prosessilla ja suunnittelun painottamisella aikaisempaan vaiheeseen hanketta (Davis 2013, p. 33).*

Davis esittää paradoksin MacLeamy-kuvaajasta. Kuvaajan mukaan suunnittelu tulisi tehdä mahdollisimman aikaisessa vaiheessa. Tämän pitäisi edesauttaa suunnittelijoita tekemään suunnittelupäätöksiä silloin, kun kustannuksia ei ole vielä sidottu ja niiden kustannusvaikutukset ovat suuret. Tällöin suunnitelmat ovat mahdollisimman pitkälle jalostettuja. Ongelmaksi muodostuu suunnittelijan mahdollisuus tehdä suunnitelmiin muutoksia hankkeen myöhäisemmässä vaiheessa. Suunnitelmat ovat pääpiirteittäin lukittu jo varhaisessa vaiheessa, mutta muutoksia ja ongelmakohtia on silti odotettavissa. Koska suurin osa isoista suunnittelupäätöksistä on jo lukittu, niitä ei voida enää muuttaa. Jotkin ongelmat tai muutokset vaatisivat kuitenkin isojakin suunnitelmamuutoksia, mutta niihin ei voida enää vaikuttaa. Tämä nostaa myöhäisemmän vaiheen suunnitelmamuutosten kustannuksia todella paljon (Davis 2013, p. 34-35).

Algoritmiavusteisen suunnittelun yhtenä lähtökohtana on pyrkiä vaikuttamaan päätöksenteon ajankohtaan ja kustannusten määräytymiseen, mutta myös muutosten kustannusvaikutuksiin. Kun algoritmiavusteinen suunnittelu toteutetaan aikaisemmin esiteltyjen laatukriteereiden mukaisesti, kyetään näihin asioihin vaikuttamaan. Davis esittää teoriansa pohjalle muunnellun MacLeamy-kuvaajan kuvassa 26. Siinä esitetään algoritmiavusteisen suunnittelun vaikutukset suunnittelijoiden kykyyn tehdä suunnitteluratkaisuita tai muutoksia ja niiden kustannusvaikutuksia suhteessa projektin ajankohtaan (Davis 2013, p. 34-36).



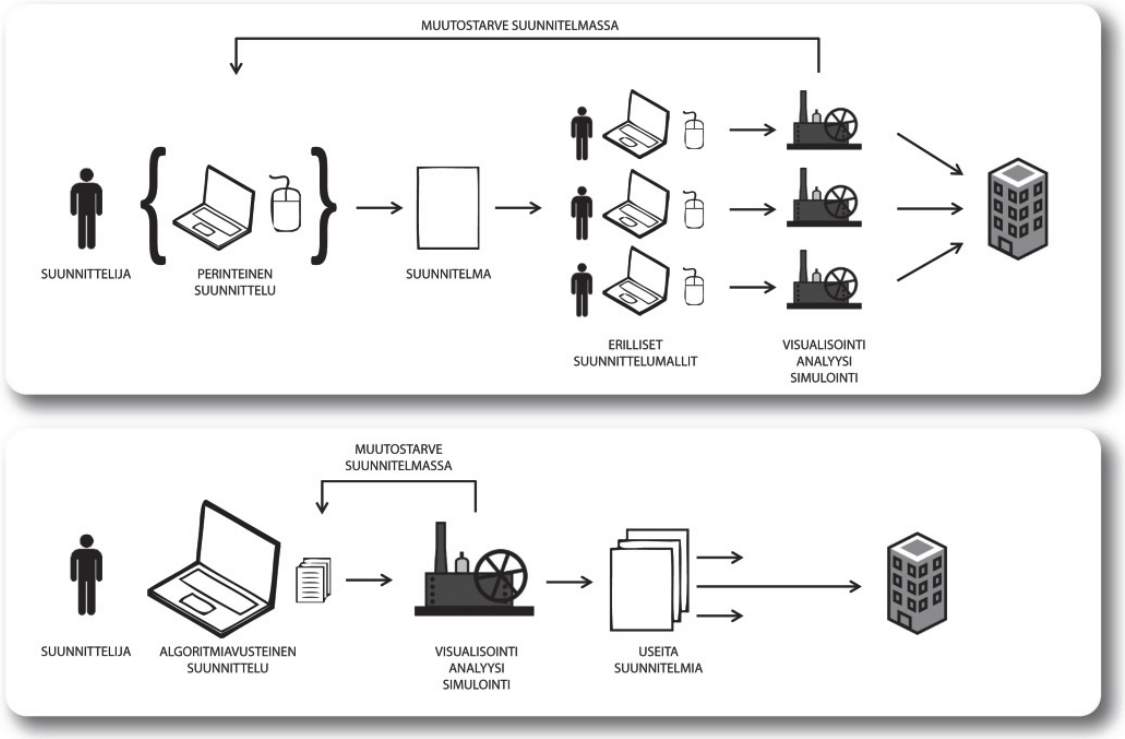
Kuva 26. *Muunneltu MacLeamy-kuvaaja algoritmiavusteisen suunnittelun vaikutuksesta (Davis 2013, p. 208).*

Kuvan 26 muunnellusta MacLeamy-kuvaajasta esitetään kaksi olennaista muutosta perinteiseen MacLeamy-kuvaajaan. Siinä esitetään, että algoritmiavusteisella suunnitteluprosessilla pystytään siirtämään kykyä vaikuttaa kustannuksiin paljon myöhäisemmässä vaiheessa. Samalla tavalla muutosten aiheuttamat kustannukset tämän algoritmiavusteisen suunnitteluprosessin aikana pysyvät matalina hyvin pitkän aikavälin. Davis kehitti tämän kuvaajan oman tutkimuksensa perusteella, mutta samaa ideologiaa esittää myös Woodbury (2010, p. 43), jossa vastoin MacLeamy-kuvaajaa suunnittelupäätöksiä tulisi lykätä myöhäisemmäksi (Davis 2013, p. 205-208).

Davis kehitti ja testasi tämän teorian Dermoid-projektin avulla. Projekti on esitetty kappaleessa 2. Projektissa tutkittiin, kuinka oikein tehdyllä algoritmiavusteisella suunnittelulla voidaan mahdollistaa esimerkiksi koko rakennuksen muodon muuttaminen tai määräytyminen myöhäisessä vaiheessa hankkeen aikataulua. Projektissa todettiin myös, että rakennuksen muotoa ei ole järkevää lukita lopullisesti ennen kuin saavutetaan suunnittelussa tilanne, jossa ymmärretään tosiasiallisesti rakennuksen muodon vaikutukset esimerkiksi kustannuksiin, aikatauluun tai rakentamiseen. Algoritmiavusteisen suunnittelun avulla on mahdollista siirtää suunnitteluratkaisut tai päätökset projektissa siihen ajankohtaan, jossa ymmärretään suunnitteluratkaisun kaikki mahdollisuudet ja lukitsemisesta seuraavat vaikutukset (Davis 2013, p. 205-208).

Tutkimuksessa on aikaisemmin esitelty Tanska ja Österlundin tutkimuksen kuva 4, jossa esitellään perinteisen suunnitteluprosessin ja algoritmiavusteisen suunnitteluprosessin käytettyä aikaa eri hankevaiheisiin nähden. Kuvaajat esittävät samaa asiaa, jota Davis tutki työssään. Kuvaajista nähdään, että luonnossuunnittelua kyetään siirtämään hyvin

pitkälle projektin aikajanalla. Tämä johtuu siitä, että algoritmiavusteisessa suunnitteluprosessissa ei tarvitse lukita merkittäviä suunnittelupäätöksiä hankkeen alussa, koska eri vaihtoehtojen ja muutosten tekeminen ei ole kustannusvaikutusten näkökulmasta mikään este (Davis 2013, p. 205-208; Tanska & Österlund 2014, p. 24-25).



Kuva 27. *Perinteisen ja algoritmiavusteisen suunnittelun eroavaisuudet (Tanska & Österlund 2014, p. 57).*

Tanska ja Österlund (2014) esittävät kuvassa 27 perinteisen ja algoritmiavusteisen suunnittelun eroja. Perinteisessä suunnittelussa suunnittelija tuottaa irrallisen suunnitelman, jonka avulla toiset suunnittelijat voivat luoda visualisointia, analysointia tai simulointia varten irralliset mallit. Näiden mallien tarkkuustaso saattaa vaihdella suunnittelijasta riippuen. Jos suunnitelmiin täytyy tehdä muutoksia, prosessi käynnistyy alusta alkaen uudestaan. Tällöin on myös käytettävä erityistä huomiota, että suunnitelmamuutokset päivitetään kaikkiin malleihin. Kaikki muutokset tehdään eri suunnittelijoiden toimesta. Algoritmiavusteisessa prosessissa suunnittelija luo algoritmiavusteisesti yhden yhteisen suunnitelman, jota voidaan suoraan hyödyntää erilaisissa visualisoinneissa, analyyseissä tai simuloinneissa. Jos muutoksia tarvitsee tehdä jonkin analysoinnin tai simuloinnin tuloksena, muutokset tehdään algoritmiseen suunnitelmaan, jolloin muutokset suoritetaan samalla näihin samoihin analysointeihin ja simulointeihin. Algoritmiavusteisen suunnittelun avulla voidaan tuottaa myös useampi vaihtoehto suunnitelmista ja verrata niitä tehokkaasti keskenään. Samalla kun algoritmiavusteisella suunnittelulla kyetään vaikuttamaan suunnitelmien kustannuksiin, kyetään myös merkittävästi vaikuttamaan suunnittelun kustannuksiin (Tanska & Österlund 2014, p. 56-57).

Yhteenvetona voidaan todeta algoritmiavusteisen suunnittelun vaikuttavan merkittävästi suunnitelmien laatuun ja hintaan. Perinteisen mallintavan ja algoritmiavusteisen suunnitteluprosessien välillä on hyvin olennaisia eroja. Tanskan ja Österlundin kuva 27 esittää, minkä takia algoritmiavusteisella prosessilla muutosten aiheuttamat kustannukset eivät ole yhtä korkeat ja työläät toteuttaa kuin ne ovat perinteisellä prosessilla. Davisin kuva 26 esittää, että algoritmiavusteisella suunnittelulla voidaan päästä tilanteeseen, jossa muutoksista aiheutuvat kustannukset ovat lähes koko suunnitteluprosessin ajan hyvin matalat ja kyky vaikuttaa suunnitelmiin on koko suunnitteluprosessin merkittävän korkea. Algoritmiavusteisella suunnittelulla voidaan siirtää merkittävien suunnittelupäätösten lukitsemista huomattavasti myöhäisempään vaiheeseen projektia, ideaalisesti siihen ajanhetkeen, kun ymmärretään kaikki merkittävät seuraamukset suunnitteluratkaisun valitsemisesta. Tämä johtaa parempaan suunnitelmien laatuun ja vähentää myöhäisen vaiheen muutoksista johtuvia kustannuksia.

5. CASE: VÄLISEINÄELEMENTIN ALGORITMIAVUSTEINEN SUUNNITTELU

Case-tutkimuksessa on tarkoituksena tutkia algoritmiavusteisen suunnittelun soveltuvuutta väliseinäelementtien suunnittelussa. Edellisessä osassa kehitettiin taustateoriaan perustuen algoritmiavusteiseen suunnitteluun prosessi, jonka toimivuutta selvitetään case-kohteella.

5.1 Tausta, lähtökohdat, kriteerit ja tavoitteet tutkimukselle

Case-tutkimuksessa hyödynnetään kehitettyä prosessia ja tutkitaan sen toimivuutta betonisten väliseinien elementtisuunnittelussa. Tarkastelussa on kohtalaisen yksinkertainen asuinkerrostalo, josta käytössä on arkkitehdin IFC-malli. Kuten edellisessä osassa todettiin, tämän työn puitteissa ei kuitenkaan tarkastella LVIAS-osuutta. Arkkitehdin IFC-mallista on kuitenkin tarkoitus erotella aukkotiedot ja etsiä keino, jolla aukkotiedot saadaan suoraan hyödynnettyä prosessiin. Tämä prosessi voisi olla myös osittain sovellettavissa LVIAS-osuuteen tulevaisuudessa.

Tutkimuksessa käytetään IFC-tiedostojen käsittelyyn SimpleBim ohjelmistoa, algoritmiavusteiseen suunnitteluun Rhinoceros 3D ja Grasshopper -ohjelmistoparia ja tietomallinnusohjelmistona Tekla Structuresta. Käytössä olevat versiot: SimpleBim 2016 versio 5.0, Tekla Structures 2016i SP 3, Rhinoceros 3D versio 5 ja Grasshopper August-27 2014 Build 0.9.0076.

Lähestytään case-tutkimuksen tavoitteita soveltuvilla osin koko tutkimukselle asetettujen tavoitteiden avulla:

- Betoniväliseinäelementin algoritmiavusteisen suunnitteluprosessin kuvaaminen
- Algoritmiavusteisen suunnittelun vaikutukset betonisten väliseinäelementtien suunnitteluprosessiin
    - Hyödyt, haitat, ongelmakohdat ja kehitystarpeet
- Ohjelmistoihin liittyvät rajoitteet
    - Rajapinnat ja ohjelmien sisäiset ongelmakohdat
- Kehittää uudelleen hyödynnettävä menetelmä algoritmin luomiseen elementtisuunnittelussa

Näiden tavoitteiden avulla tulkitaan sekä case-tutkimuksen että tulosten analysoinnin sisältöä. Case-tutkimuksessa selvitetään konkreettisesti algoritmiavusteisen mallintamisen mahdollisuutta väliseinien elementtisuunnittelussa. Tärkeimpänä tavoitteena on sel-

vittää minkälaisia kriittisiä vaiheita ja ongelmakohtia prosessissa havaitaan, koska ne voivat muodostua kriittisiksi tekijöiksi myös prosessin hyödynnettävyydelle. Huomioidaan myös tässä vaiheessa, että vaikka case-tutkimuksessa tutkitaan algoritmiavusteista suunnittelua edellä mainittujen työkalujen avulla, samalla on tarkoituksena tutkia algoritmiavusteista suunnittelua ohjelmistoista riippumattomasta näkökulmasta.

5.2 Algoritmiavusteisen suunnitteluprosessin testaus betonisten väliseinäelementtien suunnittelussa

Edellisessä osiossa luotiin olemassa olevaan tietoon ja omaan kokemukseen perustuen väliseinäelementin suunnitteluprosessi ja -kaavio. Seuraavaksi testataan prosessin toimivuutta. Edellä on esitetty kriteereitä ja mittareita, joiden avulla arvioidaan prosessin toimivuutta ja etsitään hyötyjä ja rajoitteita. Testaus on jaettu viiteen tutkittavaan osaan.

Case-tutkimuksen lähtötietona käytetään toteutuneen projektin arkkitehdin IFC-mallia. Rakennus on monikerroksinen asuinkerrostalo. Rakennejärjestelmänä toimii kantavat ulko- ja väliseinät ja ontelolaattavälipohjat. Tarkoituksena on erotella IFC-mallista kantavat betoniset väliseinät, joiden avulla määritetään mallinnettavien seinien päägeometrialinjat. Sen jälkeen luodaan algoritmi käyttäen Rhino/Grasshopper -ohjelmistoparia, jonka avulla seinät mallinnetaan Teklaan. Algoritmeilla mallinnetaan myös väliseinien elementtijako, liitokset, valutarvikkeet ja raudoitukset. Lisäksi tutkitaan minkälaisia vaihtoehtoja on elementtien valmistuskuvien tekemiseen.

5.2.1 Lähtötietojen kokoaminen ja tarkastus

Arkkitehdin IFC-mallille on asetettava vaatimuksia muun muassa siksi, että SimpleBimillä voidaan tehokkaasti erotella seinät muusta mallista. Todettakoon tässä vaiheessa case-tutkimusta, että YTV2012-kokoelmassa annettavat vaatimukset ovat riittävät, sillä niiden mukaan arkkitehdin tulisi mallintaa kaikki kappaleet käyttäen kappaleiden jäsentelyyn vaadittavia ominaisuuksia. Näitä on muun muassa nimeäminen, materiaaliominaisuus, kerros- ja/tai lohkotieto ja kantava tai ei-kantava. Todellisissa projekteissa YTV2012 on hyvä pohja yhteisille vaatimuksille, mutta projektikohtaisesti sovittavia asioita jää kuitenkin vielä paljon. Case-tutkimuksen avulla pyritään löytämään algoritmiavusteiseen elementtisuunnitteluprosessiin liittyviä vaatimuksia, jotka esitetään analysointi-osuudessa.

Aluksi arkkitehdin IFC-malli tarkastettaisiin käyttäen Solibri Model Checkeriä (Solibri). Solibri on tietomallien tarkastamiseen erikoistunut ohjelmisto. Solibrissa on olemassa valmiit YTV2012 tarkistusehdot. Tarkastuksen tarkoituksena on löytää selkeitä virheitä arkkitehdin mallista. Tämä ei ensisijaisesti ole elementtisuunnittelijan tehtävä, mutta koska Solibrilla tarkastuksen tekeminen on kohtalaisen helppoa ja tehokasta, on se järkevää tehdä. Arkkitehdin mallissa olevat virheet myöhemmässä vaiheessa havaitessa

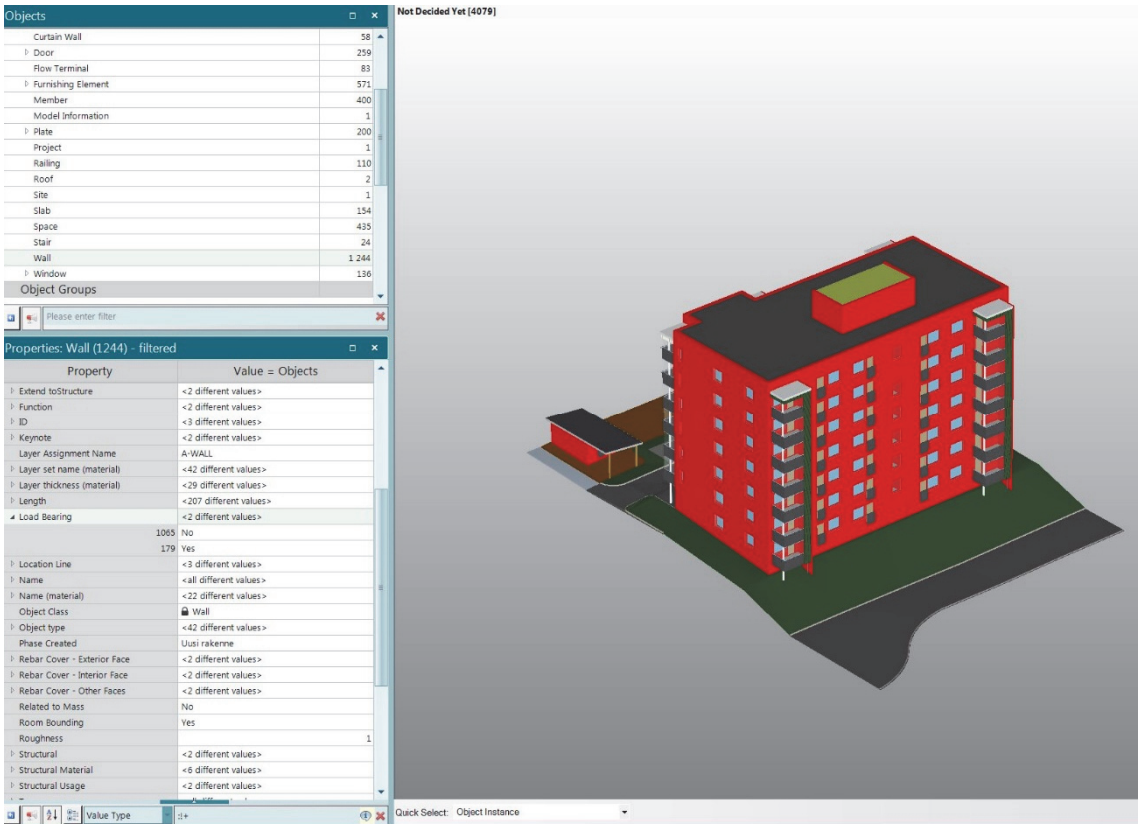
voivat aiheuttaa merkittäviä hankaluuksia elementtisuunnitteluun. Tämän tutkimuksen puitteissa Solibrilla tehtävä tarkastus jätetään kuitenkin tarkastelematta. Tulevaisuudessa olisi mielenkiintoista kehittää Solibrin tarkastuslistoja tämän tutkimuksen perusteella, jos tutkimuksella löydetään selkeitä vaatimuksia algoritmiavusteiseen prosessiin.

Lähtötietojen tarkastamiseen kuuluu myös tarkastaa, että kaikki tarvittavat lähtötiedot ovat olemassa ja saatavilla. Käytännössä se tarkoittaa sitä, että elementtisuunnittelijalla on käytössään ainakin rakennesuunnittelijan, arkkitehdin ja LVIAS-suunnittelijoiden suunnitelmat. Pieneltäkin vaikuttavan lähtötiedon puutteellisuus tai puuttuminen voi vaikuttaa todellisuudessa jonkin rakenteen suunnitteluun merkittävästi.

5.2.2 Väliseinien erottelu IFC-mallista

Väliseinien erotteluosuudessa on tarkoituksena käsitellä arkkitehdin IFC-mallia niin, että siitä erotellaan halutut rakennusosat. Tässä tapauksessa se tarkoittaa kantavia betonisia väliseiniä.

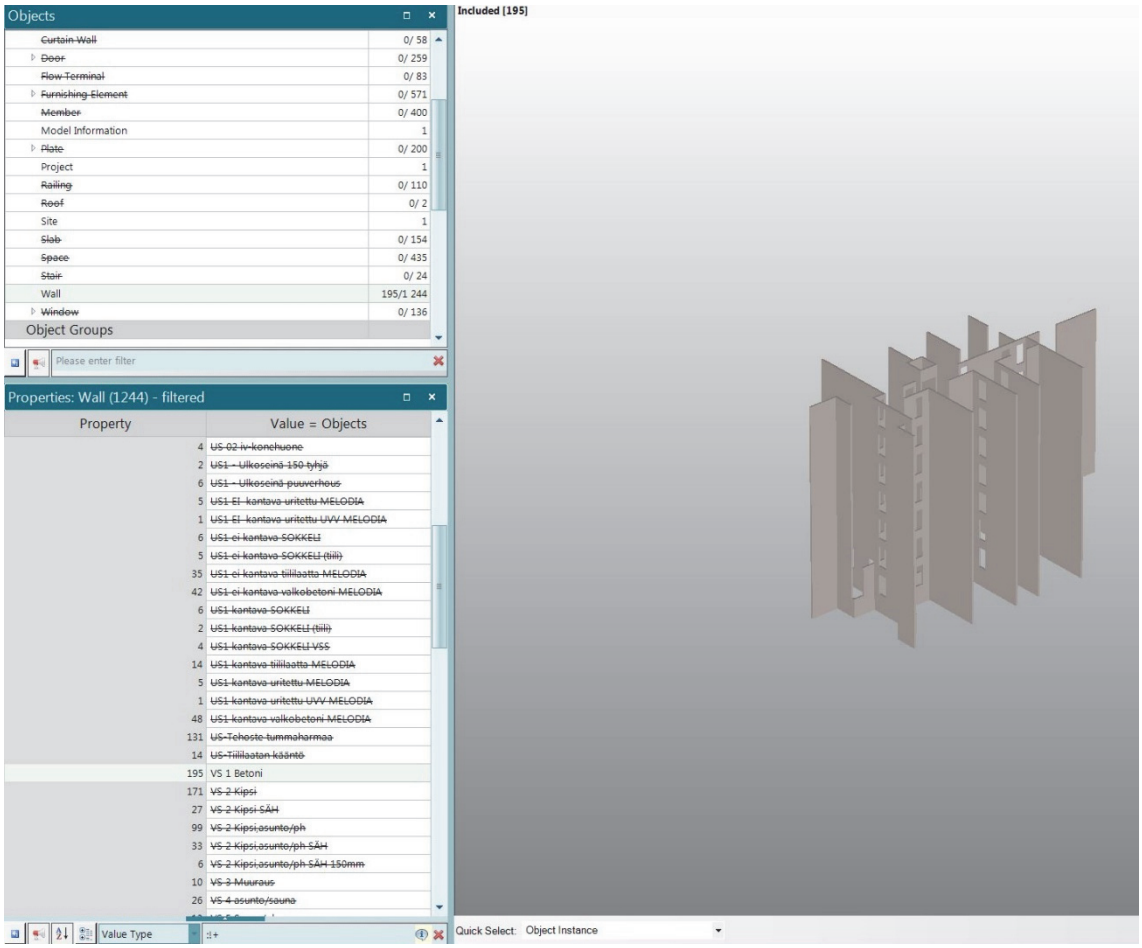
Arkkitehdin IFC-malli viedään SimpleBim-ohjelmistoon käsiteltäväksi. SimpleBim on tarkoitettu IFC-mallien käsittelyyn ja jäsentelyyn. Sitä voidaan käyttää IFC-mallin muokkaamiseen, rajaamiseen tai uudelleenjäsentelyyn. Tässä työssä sitä käytetään ainoastaan erottelemaan kantavat betoniset väliseinät muusta IFC-mallista ja luomaan uusi IFC-malli näistä seinistä.



Kuva 28. *SimpleBim näkymä ja filtteröintiominaisuudet.*

Haluttujen ja ei-haluttujen osien rajaus SimpleBimissä tapahtuu kuvien 28 ja 29 esittämillä tavoilla. Rajaus voidaan tehdä myös raahaamalla rakennusosat yksi kerrallaan pois, mutta tehokkain ja laadunvarmistuksen näkökulmasta parempi keino on käyttää osien ominaisuuksia filtteröimään kappaleita pois. Tällöin on hyvin tärkeää, että arkkitehti on mallintanut kaikki kappaleet sovittujen vaatimusten mukaisesti.

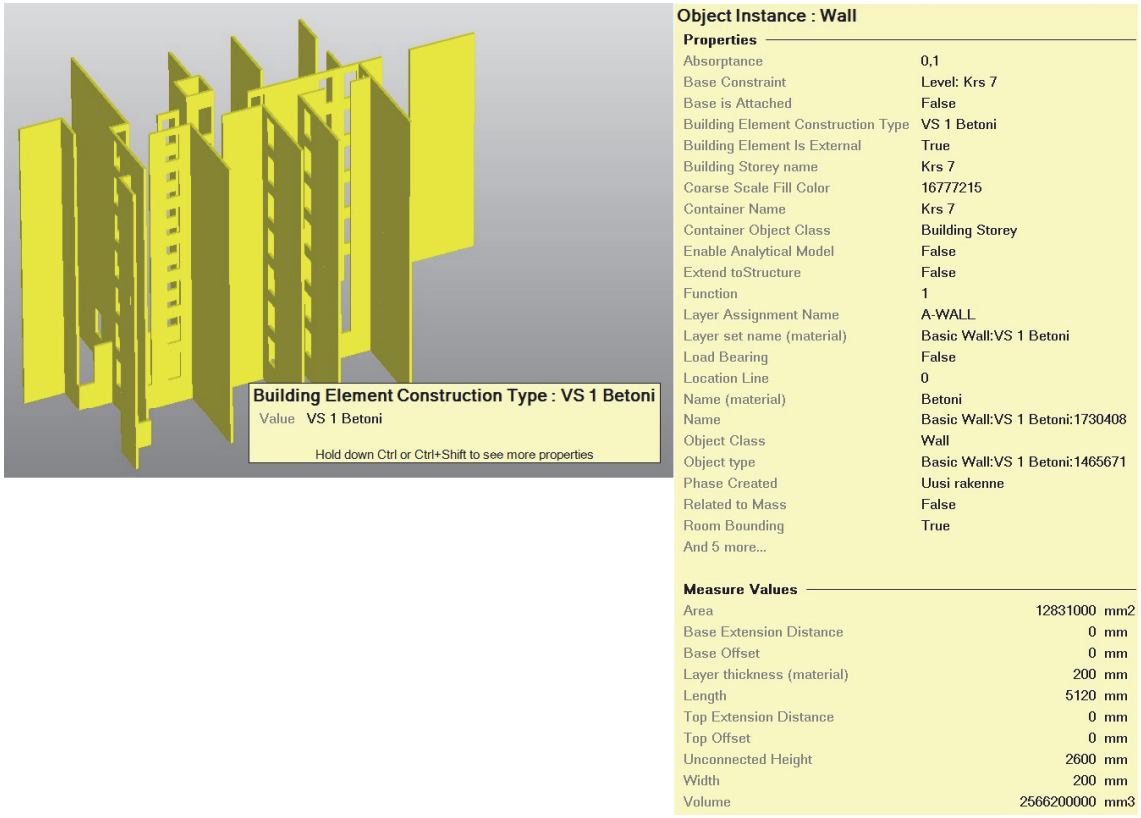
Kuvassa 28 esitellään SimpleBim näkymä ja kappaleiden filtteröinti. Tarkoituksena oli käyttää load bearing ja väliseinä -ominaisuuksia erottelemaan oikeat seinät, mutta arkkitehdin IFC-mallissa väliseinille ei oltu määritetty load bearing -ominaisuutta oikein. Projektin alussa arkkitehdille olisi tullut antaa vaatimukseksi määrittää kaikille seinille load bearing -ominaisuus. Oikeassa projektissa väärin määritettyjen ominaisuuksien takia IFC-malli olisi palautettu arkkitehdille korjattavaksi. Case-tutkimuksessa näin ei voitu tehdä, joten etsittiin toinen keino erotella kantavat betoniset väliseinät arkkitehdin IFC-mallista.



Kuva 29. *Building element construction type -ominaisuuden hyödyntäminen väliseinien erotteluun.*

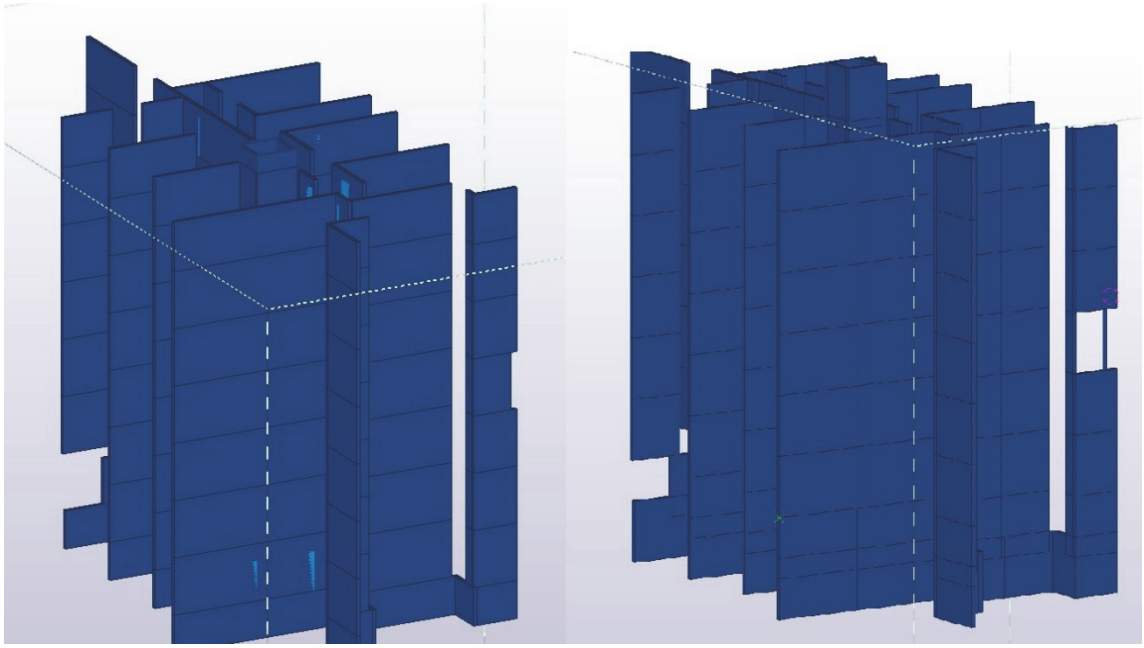
Väliseinien erotteluun löydettiin kokeilemalla toinen käyttökelpoinen ominaisuus, building element construction type. Sen ominaisuuden avulla löydettiin suoraan käyttökelpoinen määritelmä VS 1 Betoni, jolla saatiin eroteltua halutut väliseinät. Väliseinät voidaan nyt viedä SimpleBimistä omaksi IFC-tiedostoksi. SimpleBim sisältää ominaisuuden, jonka avulla voidaan luoda jokaisen kerroksen erotelluista kappaleista oma IFC-malli. Erotelluista seinistä voi tehdä myös kokonaisuudessaan IFC-mallin, sillä IFC-malli sisältää seinien korkotiedot. Case-tutkimuksessa tehtiin molemmat vaihtoehdot siltä varalta, että törmättäisiin johonkin odottamattomaan ongelmaan.

SimpleBimillä tehtiin visuaalinen ja tietosisällön tarkastukset erotelluille väliseinille. Yksinkertaisuudessaan tarkastettiin, että kaikki erotellut seinät kuuluivat valittuun filtteriin. Tarkastettiin myös, että kyseinen filtteri erotteli ainoastaan halutut väliseinät. Kuvassa 30 on esitelty suoritettu tarkastus. Seinien tarkemmista ominaisuuksista todetaan, että seinät sisältävät oikeat asiat, kuten kerrostiedon, materiaalin ja building element construction type.



Kuva 30. *SimpleBimillä tehty tarkastus erotelluille väliseinille.*

SimpleBimillä suoritettiin myös visuaalinen tarkastus virheiden varalta erotelluille väliseinille. Tämän tarkastuksen avulla voidaan havaita suuremmat virheet, kuten väärät kappaleet tai mallinnusvirheet. SimbleBim visuaalisuus ei ole kuitenkaan paras mahdollinen, joka hieman hankaloitti visuaalista tarkastusta. Kuvassa 31 esitetään Teklassa arkkitehdin mallinnusvirheestä johtuva virhe. SimpleBimillä tätä virhettä ei havaittu, koska virhe johtui siitä, että arkkitehti oli mallintanut virheellisesti seinän pintaan pienen loven. Näin pientä virhettä ei voida käytännössä SimpleBimillä havaita.



Kuva 31. *Vasemmalla korjatun algoritmin ja oikealla alkuperäisen algoritmin lopputulokset Teklassa, joista on havaittavissa arkkitehdin mallinnusvirheen vaikutukset.*

Arkkitehdin virheestä johtuva mallinnusvirhe Teklassa muodostui siitä, että Grasshopperin algoritmi tulkitsi loven seinän nurkkapisteeksi. Se synnytti seinään 8 uutta nurkkapistettä. Seinän päälinjan muodostamisen algoritmi poimii tietyt kaksi nurkkapistettä seinistä, joiden mukaisesti seinät mallinnetaan. Virheellisessä seinässä nurkkapisteitä on 8 enemmän kuin muissa ja ne ovat eri järjestyksessä, jolloin algoritmi valitsee väärät nurkkapisteet. Kuvan 32 oikeanpuoleisessa algoritmissa mallinettiin seinät ulkoreunan mukaan ja vasemmanpuoleisessa kuvassa keskilinjan mukaan, mutta ongelma säilyy joka tapauksessa.

5.2.3 Väliseinien geometrialinjojen muokkaus ja algoritmiohjattu mallintaminen

Tässä vaiheessa case-tutkimus eteni tilanteeseen, jossa alettiin muodostaa Grasshopperilla algoritmia. Algoritmin kirjoitusprosessista tuli hyvinkin pian iteratiivinen, virheen ja ratkaisun ketju. Tämä johtui siitä, ettei tutkijalla ollut kokemusta perinteisestä tai visuaalisesta koodauksesta. Lähtökohtaisena ideana algoritmille oli, että Rhinoon tuodaan väliseinät IFC-muodossa kerros kerrallaan ja seinät suunnitellaan ja mallinnetaan kerroksittain.

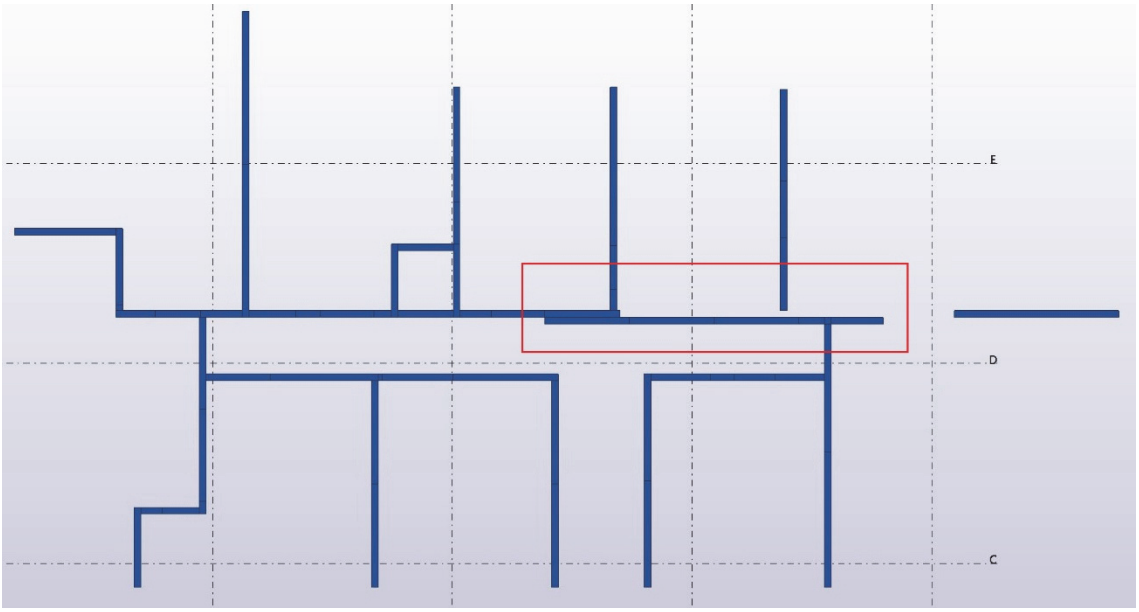
Algoritmin luomisen aikana törmättiin käytännön koodaamiseen ja ohjelmistojen tuntemattomuudesta johtuviin ongelmiin. Vaikka algoritmin luomisesta ei ollut kokemusta, teoriaosuuden tutkimuksesta sai hyvät lähtökohdat algoritmin luomisprosessiin. Algoritmin tarkoituksena on ohjata tietty tehtäväsarja koodikielen avulla alusta loppuun. Koska tiedettiin mitä haluttiin tehdä, määritettiin algoritmin tehtäväsarja sanalliseen

muotoon. Tarkoituksena oli luoda algoritmi, joka määrittää IFC-tiedostosta väliseinät, niiden päägeometrialinjat, muokkaa päägeometrialinjoja tarvittaessa, mallintaa Teklaan väliseinät ja luo väliseinien elementtijaon. Tällä pohjalla tehtäväsarja paloiteltiin erikseen hallittaviin osiin.

Tekla, Rhino ja Grasshopper -ohjelmistojen käytössä on muutamia käytännön asioita, joita tulee huomioida, jotta Grasshopper-Tekla Live Link -lisäosa toimisi. Teklamalli on oltava auki ensimmäisenä, sen jälkeen avataan Rhino ja kirjoitetaan komentokenttään Grasshopper, jolloin Grasshopper käynnistyy.

Ensimmäisessä osassa algoritmia tuotiin IFC-tiedostomuodossa muusta arkkitehtimallista erotellut väliseinät Rhinoon. Tätä tiedonsiirtoa varten täytyi asentaa GeometryGym Rhino3D/Grasshopper IFC -lisäosa. Lisäosan avulla IFC-malli tuotiin Rhinoon. IFC-malli toimii ikään kuin referenssinä Rhinossa, johon Grasshopperissa pystytään referoimaan. Tämä mahdollisti sen, että Grasshopperilla voidaan käyttää breb-nimistä kappaletta, johon referoidaan IFC-mallin seinät. Näin saatiin muutettua IFC-mallin seinät Rhinolla ja Grasshopperilla hyödynnettävään muotoon.

Kuten mainittu, alun perin tarkoituksena oli tuoda seinät kerros kerrallaan ja luoda jokaiselle kerrokselle oma algoritmi, jolloin kerroskorkeus saadaan helpoiten tehtyä. Tässä tehtiin kuitenkin ajatusvirhe, sillä kerroksen sisällä saattaa olla useamman korkuisia seiniä. Tämä teki algoritmista hyvin aikaisessa vaiheessa rajoittuneen ja ei-taipuisan. Virhe huomattiin, kun seinät oli jo mallinnettu Teklaan ja Teklaan tuotiin IFC-malli referenssiksi, johon mallinnettuja seiniä verrattiin. Tässä algoritmissa seinät mallinnettiin myös käyttäen reunalinjoja. Arkkitehdeille mallinnussuunta väliseinissä ei ole merkittävä, mutta koska algoritmi hyödynsi väliseinien reunalinjoja mallinnuslinjoina, aiheutti väärä mallinnussuunta seinien väärän sijainnin kuvan 32 mukaisesti.



Kuva 32. *Arkkitehdin mallinnussuunnasta johtuva virhe alkuperäisellä algoritmilla.*

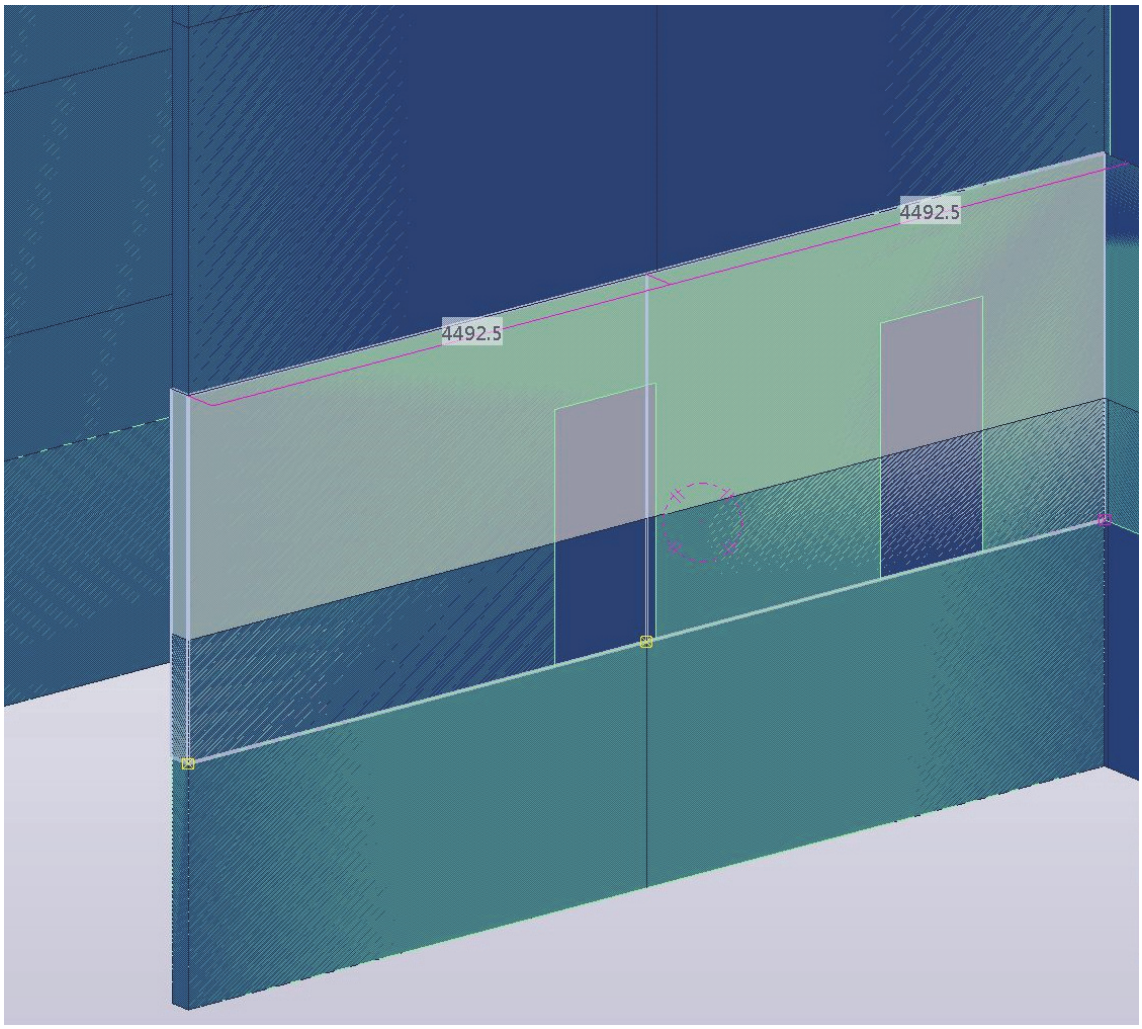
Virheistä oppineena rakennettiin algoritmi uudestaan. Tällä kertaa tuotiin kaikki IFC-mallin seinät kerralla Rhinoon. Samalla tavalla luotiin IFC-mallin seinistä breb:ejä. Algoritmiin tehtiin lisäys, joka mahdollistaa myös vinojen seinien mallintamisen. Toisin sanoen tehtiin eräänlainen koordinaatiston korjaus jokaiselle seinälle, jossa seinien suunta määritettiin vektoreiksi. IFC-malli sisältää korkotiedon seinistä ja kerroskorkeuden, joten seinien geometrialinjat projisoitiin nollatasolle. Lähtökohtaisesti brebin hajottamalla saadaan vain pintoja, nurkkapisteitä ja reunoja. Edellisestä kokeilusta ja normaaleista mallinnuskäytännöistä johtuen algoritmia kehitettiin niin, että saatiin kaikkien seinien keskilinjat päägeometrialinjoiksi. Tämän algoritmin rinnalle luotiin toinen algoritmi, joka hyödynsi alkuperäisten brebien korkotietoa, josta saatiin seinien alapintojen z-koordinaatit. Tätä tietoa hyödyntämällä siirrettiin seinien päägeometrialinjat oikeille korkotasoille.

Edellisellä algoritmilla kohdattiin ongelma, jossa kerroksen sisällä saattoi olla eri korkuisia seiniä. Uuteen algoritmiin luotiin osa, joka tarkastelee kunkin seinän brebiä ja hajottaa sen osiin. Näin saadaan brebin jokainen nurkkapiste erotelluksi. Nurkkapisteistä valitaan alapinnasta ja yläpinnasta yhdet pisteet, jotka edelleen hajotetaan, jotta saadaan selville niiden z-koordinaatti. Näiden kahden pisteen itseisarvoisesta erotuksesta saadaan seinän korkeus. Tällä tavalla määritettiin jokaiselle seinälle arkkitehtimallin mukainen korkeus.

Elementtijaon alustava hahmottelu saatiin tehtyä jo ensimmäisellä algoritmilla, ennen kuin ymmärrettiin algoritmilla olevan väärät lähtökohdat. Tällä algoritmilla saatiin väliseinät jaettua tietyn maksimipituuden mukaisesti elementteihin. Samaan aikaan tehtiin havainto, että seinien algoritmiavusteisessa elementoinnissa on useita haasteita, sillä seinien elementointiin vaikuttaa moni asia. Alla oleva listaus elementtijakoon vaikutta-

vista asioista perustuu case-tutkimuksesta saatuun tietoon ja Rambollin ohjausryhmän ja Rambollin muiden elementtisuunnittelijoiden kanssa käytyihin keskusteluihin:

- Pyritään minimoimaan elementtien lukumäärä
    - Kustannussäästöt
        - Suunnittelu
        - Valmistus
        - Kuljetus
        - Asennus
        - Materiaali
    - Aikasäästöt
        - Suunnittelu ja kuvatuotanto
        - Valmistus
        - Kuljetus
        - Asennus
- Arkkitehtoniset seikat
    - Saumojen piilottaminen
    - Esteettiset seikat
- Elementin paino ja koko
    - Kuljetuksen rajoitteet
    - Asennus tulee olla mahdollista
    - Tehtaan valmistustapa ja tehdasnostureiden rajoitteet
- Pystysaumojen tulisi olla samoissa kohdissa päällekkäisillä seinillä
- Ovi- ja muut aukot
- Talotekniikan varaukset
- Haastavat elementtien muodot
    - Painopiste
- Kohtisuorasti liittyvät seinät
    - Pyritään piilottamaan seinien elementtijako
        - Kolmen seinän liitoskohdat
        - Ulkoseinien ja väliseinien välisissä liitoksissa sama tilanne

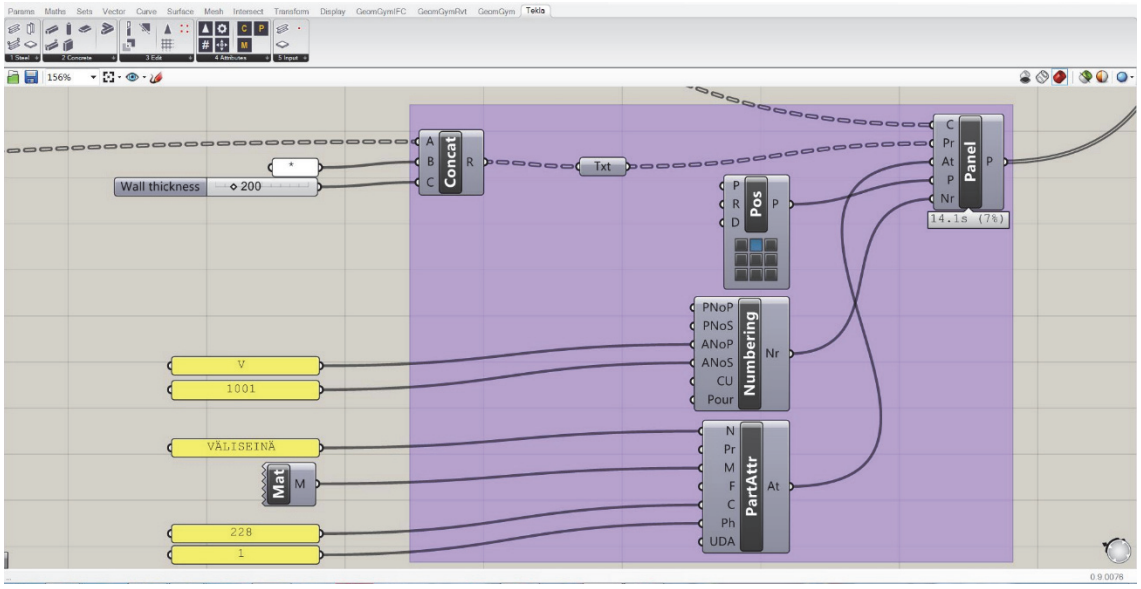


Kuva 33. *Tekla- ja arkkitehdin IFC-mallin törmäytys Teklassa.*

Kuvassa 33 on alkuperäisellä algoritmilla luotu tilanne Teklassa, jossa elementtijako saatiin hahmoteltua yksinkertaisesti. Teklaan on tuotu arkkitehdin betonisista väliseinistä luotu IFC-malli referenssiksi. Todetaan kuvasta kolme ongelmaa: alimman kerroksen seinien korkeus tulisi olla pienempi, elementtien välinen sauma risteää oviaukon kanssa ja päällekkäisten seinien pystysaumat eivät kohtaa keskenään. Ensimmäinen ongelma ratkaistiin uuden algoritmin avulla. Kahta muuta ongelmaa ei saatu ratkaistua. Edellä esitetyn elementtijaon suunnitteluun liittyvän listauksen perusteella todetaan, että on hyvin hankalaa luoda sellainen algoritmi, joka kykenisi ottamaan yksiselitteisesti kaikki listan asiat huomioon. Yksinkertainen elementtijako voidaan algoritmiavusteisesti toteuttaa, mutta algoritmin luominen, joka sisältäisi kaikki edellisessä listassa olevat asiat, olisi äärimmäisen hankala luoda.

Tässä vaiheessa case-tutkimusta ollaan tilanteessa, jossa tutkimuksen kannalta on olemassa kaikki tarvittavat tiedot seinien mallintamiseen Teklaan. Numerointi- ja ryhmittelytiedot on sovittu yhdessä kaikkien osapuolten kesken projektin alussa. Kerrostiedon eli starting number -ominaisuuden käyttäminen algoritmisesti olisi tavoiteltava tilanne. Tutkimuksen puitteissa tämä jätetään kuitenkin kokeilematta, koska niiden lisääminen

tietomalliin on kohtalaisen tehokasta jo nykyisellä tavalla. Seinien mallintamista varten tarvittiin Grasshopper-Tekla Live Link -niminen lisäosa, joka on Trimblen kehittämä. Lisäosa tuo Grasshopperiin lisätyökaluvalikon yläpalkkiin, missä on useampia mallintamiseen ja detaljointiin liittyviä komponentteja. Tutkimuksessa valittiin käyttää panel-komponenttia, joka on tyypillisesti väliseinien mallintamiseen käytettävä komponentti.



Kuva 34. *Grasshopper-Tekla live link -lisäosa ja panel-komponentin algoritmi*

Kuvassa 34 yläreunassa on Grasshopper-Tekla Live Link -lisäosa näkyvillä. Oikealla canvaksella on panel-komponentti, jonka avulla seinät mallinnetaan Teklaan. Ylimpään inputtiin tulee seinien päägeometrialinjat ja seuraavaan profiili. Profiili muodostuu seinän korkeudesta ja paksuudesta. Kuten aikaisemmin mainittu, seinän korkeus on jokaiselle seinälle erikseen määritetty algoritmilla, mutta seinien paksuus määräytyy kiinteällä parametrilla. Attribuutti- ja numerointiedot määritetään kahdella seuraavilla inputeilla. Seinien materiaali määritetään lisäosan tuomalla materiaali komponentilla. Komponentti hakee avoinna olevasta Teklasta materiaalikirjaston, josta valitaan haluttu materiaali. Vastaavasti materiaali C30/37 voitaisiin kirjoittaa tekstikomponentilla.

Huomion arvoinen seikka kuvassa 34 on panel-komponentin suorittamiseen käytettävä aika, 14 sekuntia. Yleisesti ottaen Grasshopperin komponenttien suoritusaika vaihtelee kymmenien ja satojen millisekuntien luokassa, muutamia poikkeuksia lukuun ottamatta. Todettakoon panel-komponentin käyttämän ajan olevan kohtalaisen suuri verrattuna muihin komponentteihin.

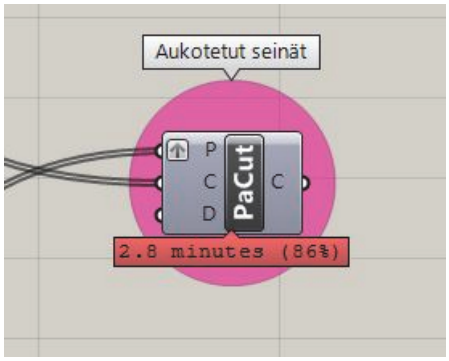
5.2.4 Väliseinien algoritmiohjattu detaljointi

Elementtijakoon liittyvistä ongelmista huolimatta jatkettiin algoritmiavusteisen prosessin tutkimista. Prosessin mukaisesti detaljointi aloitetaan väliseinien aukottamisella. Arkkitehdin IFC-mallissa aukot ovat mukana, joten ne ovat myös Rhinossa. Aukotuk-

sen periaatteena on luoda arkkitehdin IFC-mallin mukaisesti oviaukoista kappaleet, suurentaa aukkoja kymmenen milliä seinän paksuuden suuntaisesti molemmin puolin mallinnusteknisistä seikoista johtuen, mallintaa aukkokappaleet Teklaan ja käyttää Grasshopperissa Teklan part cut -työkalua.

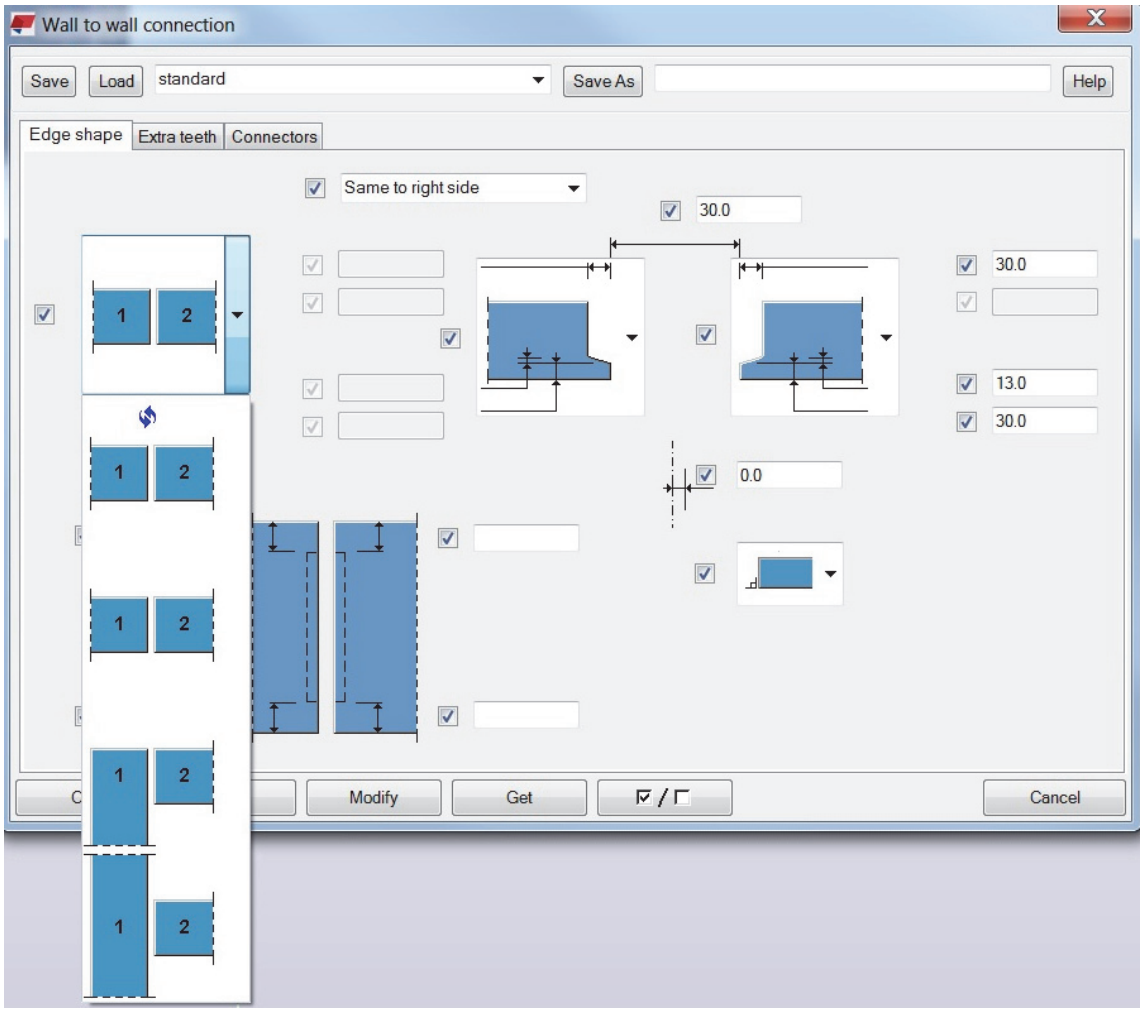
Aukkojen mallintaminen alkaa samoista brebeistä, kuin itse seinät. Brebeistä erotellaan seinät, jotka sisältävät aukkoja. Aukkoja sisältäville seinille tehdään sama koordinaatiston muunnos, kuin seinien mallintamisessakin. Tämän jälkeen seinistä haetaan nurkkapisteet. Nurkkapisteistä valitaan oviaukon pisteet, joiden avulla luodaan oviaukon pinnat. Näitä pintoja venytetään vaaditun verran, jolloin saadaan huomioon asennustoleranssit ja mallinnustekninen seikka aukkotyökalua varten. Aukkotyökalua varten leikkaava kappale tulee olla suurempi kuin leikattava kappale, eli niiden pinnat eivät saa leikata toisiaan. Toinen osa algoritmia hakee oviaukkojen korkeuden. Lopuksi käytetään slab-työkalua mallintamaan oviaukot Teklaan.

Part cut -työkalua varten tarvitaan leikattavat ja leikkaavat kappaleet. Yksinkertaisuudessaan komponentissa pääosana eli leikattavana on väliseinät ja leikkaavana osana oviaukkokappaleet. Kuvassa 35 esitetään part cut -työkalu. Siinä myös esitetään, että tämän komponentin suorittaminen kestää noin kolme minuuttia. Aikaa voidaan verrata panel-komponentin käyttämään 18 sekunnin aikaan, jolla mallinnettiin väliseinät Teklaan. Jo näiden välillä ero on suuri, mutta jos kolmea minuuttia vertaa muiden komponenttien muutamiin sekunninkymmenyksiin, on ero valtava.



Kuva 35. *Part cut -työkalu ja sen käyttämä aika.*

Detaljoinnissa siirryttiin prosessikaavion mukaisesti suunnittelemaan liitoksia ympäröiviin rakenteisiin. Käytännössä se tarkoitti vierekkäisten väliseinäelementtien välisiä pystysaumoja ja päällekkäisten väliseinäelementtien välisiä vaakasaumoja. Käsitellään ensiksi vierekkäisten elementtien väliset pystysaumat.



Kuva 36. *Vierekkäisten betonisten väliseinäelementtien välinen pystysauman liitoskomponentti Teklassa.*

Pystysaumojen liitosten kanssa kohdattiin ongelmia. Ongelman demonstroimiseksi hyödynnetään kuvaa 32 ja kuvaa 36. Kuvassa 32 on esitetty rakennuksen väliseinien geometria. Sen avulla nähdään myös että väliseinien välillä on useita nurkkatyyppejä. On seiniä, jotka eivät liity mihinkään seinään. Sen lisäksi on seiniä jotka liittyvät yhteen tai kahteen muuhun väliseinään. Kuvasta 36 nähdään, että kahden väliseinän liitokset voivat olla päittäisliitoksia tai 90 asteen nurkkaliitoksia. Nurkkatyypeissä on myös väliä, miten päin liittyvät seinät ovat toistensa suhteen. Liitoskomponentin näkökulmasta on eri asia, kumman seinän kylkeen toinen seinä liittyy. Näiden asioiden lisäksi tulisi huomioida kumpaan suuntaan nurkka kääntyy, eli mistä suunnasta elementtisaumojen juottaminen tehdään. Komponentin inputteina grasshopperissa tulee olla main part ja secondary part. Nämä kaksi inputtia määräytyvät juurikin edellä listattujen asioiden mukaisesti.

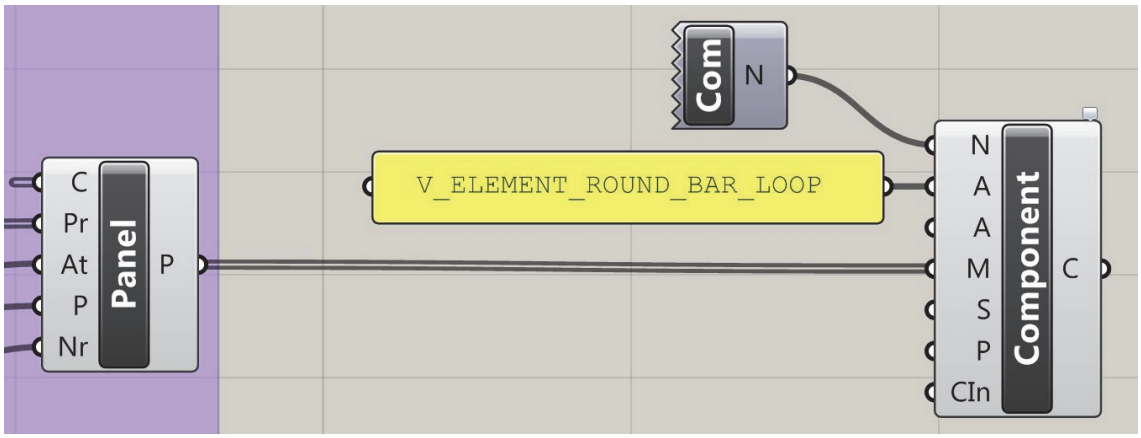
Tutkija on tehnyt kaksi vuotta elementtisuunnittelua, jonka aikana kyseessä olevaa liitostyyppiä ja -komponenttia on käytetty useasti. Kahden vuoden aikana tutkija on todennut, että tämän komponentin täydellisesti oikealla tavalla käyttäminen on hyvin

haastavaa. Case-tutkimuksessa yritettiin luoda algoritmia, joka kykenisi tunnistamaan eri nurkkatyypit, mutta algoritmin haastavuuden ja ajanpuutteen takia onnistunut lopputulos jäi selkeästi saavuttamatta.

Toinen liitostyyppi on päällekkäisten väliseinien välinen vaakasauma. Teklakomponentti tämän liitoksen tekemiseksi vaatii main partin, secondary partin ja kaksi pistettä, jotka määräävät liitoksen pituuden ja z-koordinaatin. Komponentin luomisen monimutkaisuus luo välittömästi haasteita. Grasshopperissa oleva data muuttuu hankalaksi käsitellä, kun seinät mallinnetaan Grasshopperin avulla Teklaan. Panelkomponentin output data muuttuu käytännössä vain Teklan komponenttien ymmärtämään muotoon. Tämä tarkoittaa sitä, että esimerkiksi seiniä voidaan käyttää toisessa Teklan komponentissa Grasshopperissa, mutta muut Grasshopperin komponentit eivät osaa käsitellä seinien outputtien dataa. Tästä johtuen vaakasauman komponentin vaatimien valintojen syöttäminen on todella haastavaa, eikä myöskään tämän komponentin onnistuneeseen käyttämiseen liittyviä ongelmia saatu ratkaistua tutkimuksen puitteissa.

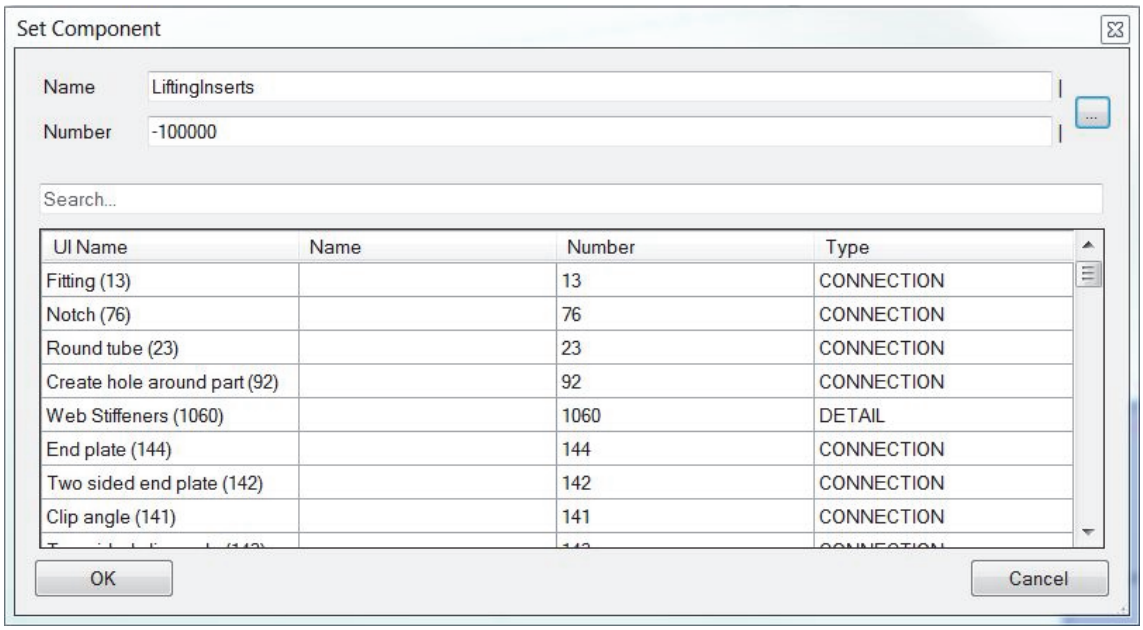
Case-tutkimuksessa siirryttiin seuraavaan vaiheeseen: valutarvikkeet ja muu detaljointi. Tämä osa käsittää väliseinien tapauksessa yleensä nosto-osat ja tönäreitä varten sisäkierreosat. Teklassa näiden komponenttien mallintaminen on kohtalaisen yksinkertaista. Nosto-osa -komponentilla määritetään halutun tyyppinen nosto-osa, jonka jälkeen komponentti laskee ja mallintaa elementin painoon ja mahdollisiin muihin ehtoihin viitaten oikeat nosto-osat. Sisäkierreosat mallintava komponentti toimii myös helposti. Komponentin voidaan antaa itse määrittää sopiva korkeus osille. Osien vaakasuuntainen sijainti on suunnittelijan määritettävissä. Komponentti luo lähtökohtaisesti kaksi osaa, mutta tietyn rajapainon ylitettäessä se luo kolmannen osan elementin painopisteen kohdalle vaakasuunnassa.

Tyypillisesti Teklalla työskennellessä komponenteille halutaan asettaa projektikohtaisesti asetettuja ja hyväksyttyjä esiasetuksia. Esimerkiksi sisäkierreosien komponentille voitaisiin määrittää tietty korkeus osien sijoittelulle ja tietty reuna-etäisyys, joka toistuisi täten kaikissa elementeissä. Tällöin komponentin esiasetukset tallennetaan komponentin asetusvalikkoon. Grasshopper-Tekla Live Link komponentit pystyvät viittaamaan näihin esiasetuksiin.



Kuva 37. *Grasshopperin Tekla komponentti, johon on asetettu nostolenkkikomponentti ja esiasetukset.*

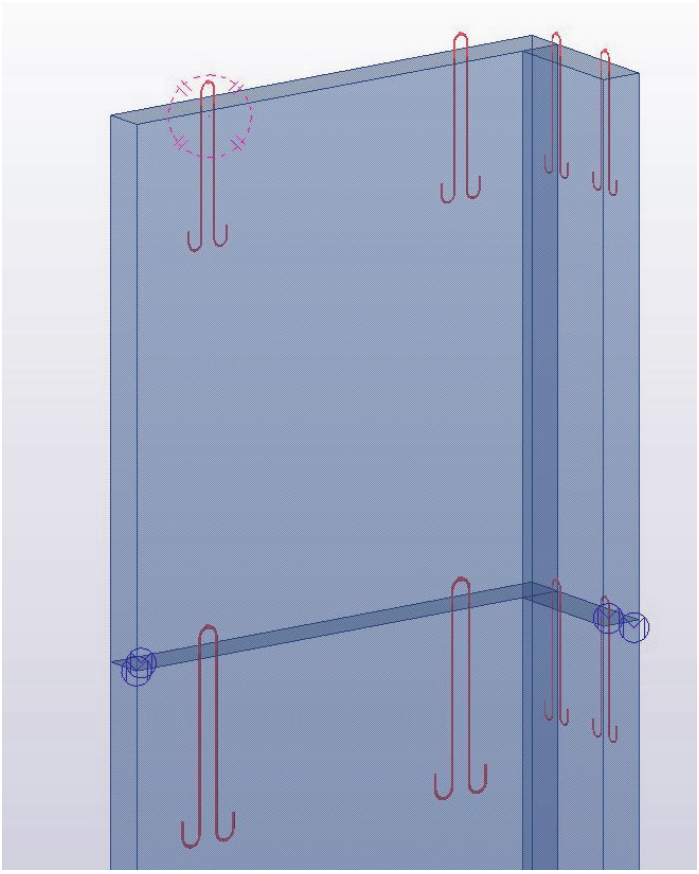
Kuvassa 37 esitetään Grasshopperin lisäosan Grasshopper-Tekla Live Linkin tuoma Tekla komponentti -komponentti. Tällä komponentilla voidaan käyttää kaikkia Teklan komponenttikatalogin komponentteja. Ylimpään inputtiin tulee komponentin nimi. Com-niminen komponentti on suora linkki avoinna olevaan Tekla-malliin. Seuraavaan inputtiin tulee esiasetuksen nimi. M-inputtiin eli mainpart-inputtiin tulee halutut seinät. Nosto-osa -komponenttia käytettäessä tarvitaan vain mainpart, joten tämän enempää inputteja ei tarvitse käyttää.



Kuva 38. *Component komponentin valintaikkuna*

Kuva 38 esittää Com-komponentin valikkoa, josta valitaan haluttu komponentti. Teklassa nosto-osan komponentille eli lifting inserts -komponentille on asetettu esiasetuksia, jotka on määritetty rakennetyypeittäin ja erityyppisten nosto-osien mukaan. Esiasetuksiin viitataan luomalla Grasshopperin panel-komponentti, johon on kirjoitettu esiasetuk-

sen nimi. Esiasetusten käyttämisellä varmistetaan, että käytetyillä komponenteilla on aina halutut asetukset.



Kuva 39. *Nosto-osat mallinnettuna Teklaan.*

Kun Grasshopperin komponentille asetetaan main partit, alkaa Tekla taustalla toteuttamaan komentoa. Mallinnetut nosto-osat esitetään kuvassa 39. Samalla tavalla sisäkierreosat ja seinien raudoitukset saatiin mallinnettua käyttäen edellä esitettyä tapaa luoda komponentti. Näiden yksinkertaisten komponenttien, joissa on yksi main part, käyttö onnistuu helposti. Kun valintoja tulee tehdä enemmän ja vaihtoehtoja komponentin asetuksille on todella useita, on algoritmin luominen todella haastavaa. Lopputuloksena case-tutkimuksella oli liitoksia lukuun ottamatta detaljoidut väliseinät, jotka sisälsivät täysin algoritmiohjatusti mallinnetut aukotukset, raudoitukset, nosto-osat ja tönärit.

Myös yksinkertaisten komponenttien mallintamisessa havaittiin ongelmia. Ongelmaksi muodostui se, ettei komponentti ladannut jokaiselle komponentille oikeita esiasetuksia. Osaan seinistä tuli standard esiasetukset, vaikka esiasetus oli valittu toisin.

5.2.5 Väliseinien algoritmiohjattu kuvatuotanto

Väliseinäelementtien valmistuskuvien toteuttamiseen on kaksi tapaa. Ensimmäinen tapa on luoda algoritmi, jonka avulla kuvat luodaan. Toinen tapa on tehdä elementtikuvat perinteisellä tavalla Teklassa. Diplomityön ohjausryhmän kanssa käytyjen keskustelui-

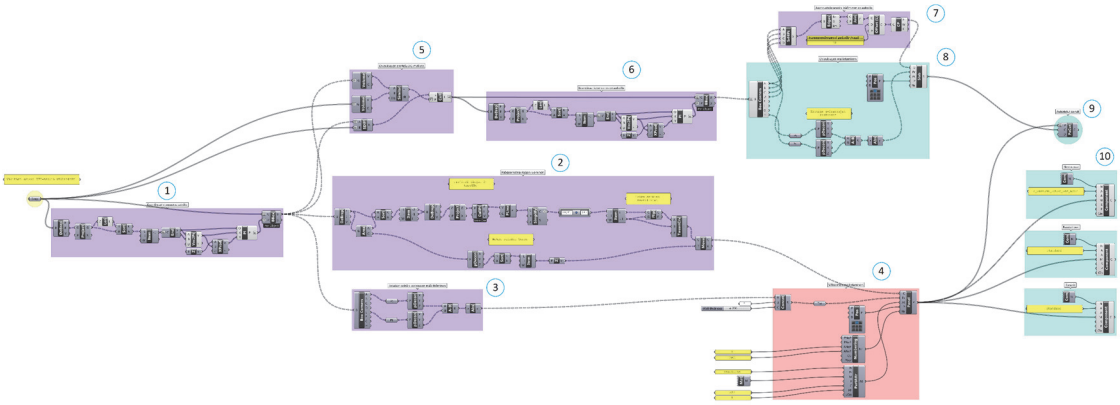
den perusteella päätettiin valita elementtikuvien luominen Teklassa. Ohjausryhmällä on kuitenkin kokemusta myös algoritmiohjatusta kuvatuotannosta, mutta sen ei koettu olevan järkevää elementtisuunnittelussa. Tärkeimpänä syynä tähän valintaan oli hyvin pitkälle kehitetyt kuvatuotannon prosessit ja kuvapohjat Teklassa, jonka takia ei ole järkevää kehittää uutta prosessia ja algoritmeja Grasshopperiin.

Kun elementit oli detaljoitu kokonaisuudessaan Teklaan algoritmin avulla, luotiin kuvat Teklassa. Tämä prosessi on täysin analoginen mallintavan elementtisuunnittelun kuvatuotantoon. Ensimmäisenä seinät ja kaikki niiden varustelut numeroidaan. Numerointitulostus tarkastetaan ja hyväksytään. Seuraavaksi valitaan haluttu kuvapohja ja luodaan kuva tämän perusteella. Elementtikuva viimeistellään lisäämällä ja muokkaamalla mittaviivoja ja muita merkintöjä kuvaan.

Esimerkki tuotetusta elementtikuvasta on esitetty liitteessä B. Tuotantovalmista kuvaa ei tehty koska nyt kuvasta näkee vielä hyvin puutteet ja virheet, jotka johtuvat algoritmista. Teklan puolella ei ole mallinnettu mitään vaan kaikki on suoraan algoritmin tulosta. Elementtikuvasta puuttuu vielä detaljointiin liittyviä seikkoja, kuten vaaka- ja pystyliitokset ja aukotuksiin liittyvät detaljit. Elementtikuvan sisällöstä puuttuvat myös: muotti- ja valupintojen merkinnät, suunnittelun lähtötiedot, tuotetiedot ja nimiötiedot. Näistä osa on sellaisia, joita ei ole mahdollista tai järkevää algoritmiavusteisesti lisätä, vaan ne on kaikkein helpointa lisätä mallin puolella. On siis hyvä huomata, että kaikkea ei ole mahdollista tai järkevää yrittää algoritmisesti lisätä. Useassakin tapauksessa on tasapainoteltava tämän asian kanssa.

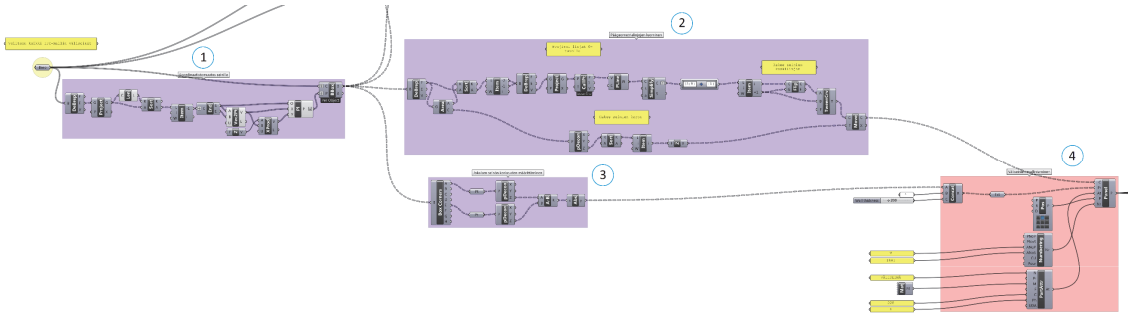
5.2.6 Case-tutkimuksessa luotu algoritmi

Teoria-osuudessa tutkittiin paljon muun muassa algoritmin uudelleenkäyttämistä, taipuisuutta, selkeyttä ja sen ryhmittelyä. Näitä opittuja asioita pyrittiin hyödyntämään algoritmia luodessa. Case-tutkimuksen algoritmin ryhmittelyyn ja luomiseen käytettiin hajauta ja hallitse -tekniikkaa. Kuvassa 40 esitetään lopputuloksen algoritmi, jolla väliseinät ja niiden detaljointi saatiin mallinnettua muutamia poikkeuksia lukuun ottamatta.



Kuva 40. *Väliseinäelementin mallintamiseen ja detaljointiin luotu algoritmi kokonaisuutena.*

Algoritmi on jaettu 10 eri osaan. Osat on värikoodattu niin, että lähtötiedot ovat keltaisena ryhmänä, päägeometrian muokkaus on purppurana ryhmänä, seinien mallintaminen punaisena ryhmänä ja detaljointiin liittyvät osiot ovat syaanin värisissä ryhmissä. Alkuperäinen hajauta ja hallitse -tekniikalla luotu tehtävän pilkkominen toimi hyvänä lähtökohtana algoritmille. Tutkijan kokemattomuus sekä koodaamisesta että visuaalisesta koodaamisesta vaikutti koodiin ja sen luomisprosessiin melko paljon. Alun perin hajauta ja hallitse -tekniikalla saadut lähtökohtaiset ryhmittelyt muuttuivat case-tutkimuksen aikana mitä enemmän tutkija oppi uusia asioita Grasshopperin ominaisuuksista.

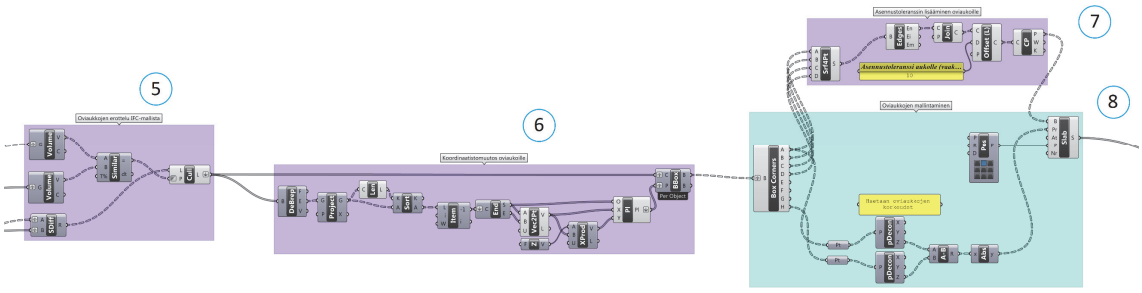


Kuva 41. *Algoritmi väliseinien mallintamiseen.*

Ensimmäisessä osa-alueessa algoritmia mallinnetaan väliseinät. Osa-alue on esitetty kuvassa 41. Algoritmi koostuu neljästä eri osasta, jotka on merkattu kuvaan numeroin. Ensimmäinen osa algoritmista tekee koordinaatistomuutoksen seinille. Tämä antaa seinien geometrialle enemmän vapauksia, sillä alkuperäisellä algoritmilla kyettiin käsittelemään vain x- ja y-koordinaattien suuntaisia seiniä. Koordinaatistomuutoksella seinistä tehdään vektoreita jotka mahdollistavat vinojen seinien käsittelyn.

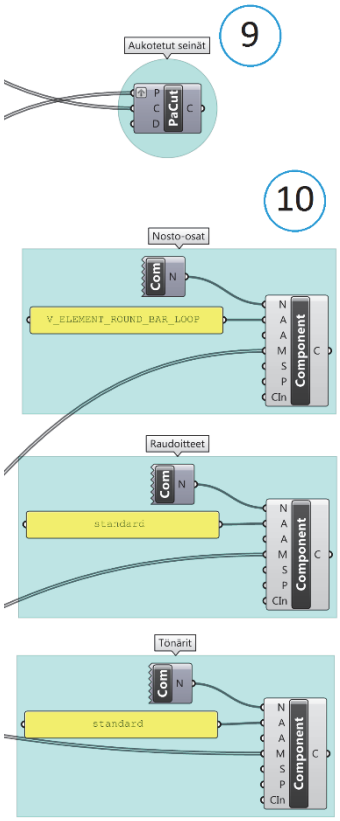
Toinen osa väliseinien mallintamisesta luo väliseinien päägeometrialinjat. Algoritmi hajottaa seinien geometrian, projisoi seinien reunalinjat 0-tasolle, hakee seinien koron,

luo reunalinjoista seinille keskilinjat ja siirtää seinien keskilinjat oikeaan korkoon. Kolmas osa hajottaa väliseinien geometriat nurkkapisteiksi, valitsee niistä ala- ja yläpinnasta yhdet pisteet, laskee niiden z-koordinaattien itseisarvoisen erotuksen ja määrittää näin jokaiselle seinälle yksilöllisen korkeuden. Neljäs osa hyödyntää aikaisimmista osista päägeometrialinjat, korkotiedot ja seinien korkeudet. Väliseinille määrättävät numerointi- ja attribuuttitiedot syötetään Teklan panel-kompontille tässä vaiheessa. Lopputuloksena tällä algoritmilla seinät on mallinnettu Teklassa.



Kuva 42. *Algoritmi oviaukkojen mallintamiseen.*

Seuraavassa osa-alueessa algoritmilla mallinnetaan oviaukot Teklaan. Osa-alue on esitetty kuvassa 42. Viides osa erottelee kaikista seinistä seinät, jotka sisältävät oviaukkoja. Kuudes osa tekee oviaukoille saman koordinaattimuunnoksen kuin osa yksi. Seitsemäs osa käyttää inputtina kahdeksannen osan box corners -komponenttia, josta valitaan oviaukoista neljä pistettä, joista muodostetaan jokaisen oviaukkokappaleen alin pinta. Pinnasta muodostetaan yhtenäinen curve, jota lopuksi venytetään asennustoleranssin verran ulospäin. Tällä tavalla saadaan ratkaistua sekä cut part -työkaluun liittyvä pintojen leikkaamiseen liittyvät että aukkojen detaljikkaan liittyvät ongelmat. Kahdeksas osa määrittää box corners -komponentin nurkkapisteiden avulla jokaisen aukon korkeuden. Lopuksi käytetään Teklan slab-komponenttia mallintamaan oviaukot Teklaan.



Kuva 43. *Algoritmi väliseinien detaljointiin.*

Kolmannessa osa-alueessa algoritmilla suoritetaan mallin detaljointia, joka on esitetty kuvassa 43. Yhdeksäs osa on cut part -komponentti, jolla väliseiniin tehdään aukot oviaukkokappaleiden mukaisesti. Tämän komponentin kanssa todettiin useita ongelmia, joista osa käsiteltiin jo aikaisemmissa osissa. Myöhemmin havaittiin, kun algoritmi saatiin luotua loppuun asti detaljoinnin osalta, että tämä komponentti on osasyyllinen mallin jumiutumiseen. Kun Tekla, Rhino ja Grasshopper avattiin uudestaan, havaittiin, että malli alkaa suorittaa algoritmia yhä uudestaan ja uudestaan. Muutaman tunnin jälkeen Tekla ja Rhino jouduttiin sulkemaan pakottamalla.

5.3 Prosessi algoritmin luomiselle elementtisuunnittelussa

Case-tutkimuksen perusteella kehitettiin yleistettävä prosessi elementtisuunnittelun rakennusosien algoritmin luomiselle. Prosessi on rajattu algoritmin luomiseen, eikä siinä oteta kantaa millä tavalla IFC-malli käsitellään tai siirretään Rhinoon. Prosessi on luotu Tekla, Rhino ja Grasshopper -ohjelmistojen yhdistelmällä, mutta tässäkin tapauksessa prosessia voidaan ajatella myös ohjelmistoista riippumattomana. Prosessikaavio on esitetty liitteessä C.

Algoritmin luominen aloitetaan tuomalla IFC-malli Rhinoon. Ensimmäiseksi IFC-mallin kappaleet on muunnettava Grasshopperin ymmärtämään muotoon. Toisin sanoen Rhinossa olevat IFC-kappaleet muunnetaan Grasshopperin kappaleiksi. Se minkälaisik-

si kappaleiksi IFC-kappaleet muunnetaan, riippuu ensinnäkin rakennusosasta ja toiseksi siitä, mitkä ovat niiden Tekla komponentin inputit. Sama voidaan yleistää muihinkin mallinnusohjelmistoihin. Algoritmiavusteisessa mallintamisessa mallinnusohjelmisto asettaa paljon vaatimuksia, joten koko prosessin ajan on tärkeä tuntea, miten Teklan mallinnuskomponentti toimii.

Seuraava osa jaetaan kolmeen saman aikaiseen osaan. Tarkoituksena on luoda kolme erillistä algoritmia. Ensimmäinen niistä hajottaa geometrian palasiksi ja muokkaa ne hyödynnettävään muotoon. Toisella haetaan kappaleiden korkoasema ja viimeisellä niiden profiilit. Kuten edellisessä osassa, tässäkin on äärimmäisen tärkeää tietää, miten kyseessä olevien rakenteiden mallinnuskomponentti toimii. Geometrian muokkaus ja käsittely ovat hyvin erilaisia, jos verrataan esimerkiksi laattaa ja pilaria. Laataston mallintamiseen vaaditaan sen nurkkapisteet, korko ja laataston paksuus. Pilarin mallintamiseen vaaditaan profiili ja ylä- ja alapäästä pisteet, joiden avulla määritetään pilarin pituus. Myös koron määrittäminen on erilaista eri rakennusosilla. Pilarin pituus määritetään sen ylä- ja alapään korkojen avulla, kun taas laatastoiden korko voidaan määrittää ala- tai yläpinnan avulla.

Tässä vaiheessa kappaleille määritetään projektikohtaisesti sovitut numerointi-, nimeämis- ja attribuuttitiedot. Kun kyseessä olevan rakennusosan mallintamiseen vaadittavat tiedot on saatu, voidaan kappaleet mallintaa Teklaan. Mallintaminen tapahtuu samalla tavalla kuin case-tutkimuksessa on esitetty. Seuraavaksi mallinnetaan sekä arkkitehdin että talotekniikan suunnittelijoiden aukotukset ja varaukset. Case-tutkimuksessa arkkitehdin väliseinäelementtien aukot saatiin algoritmiavusteisesti mallinnettua tehokkaasti. Tutkimuksen perusteella ei kuitenkaan suoraan voida todeta, että sama tapa toimii parhaiten myös muiden rakennusosien kanssa. Talotekniikan rei'itysten ja varausten mallintamiseen on olemassa tehokkaita keinoja Teklassa. Lisäksi talotekniikan suunnittelijoiden aukko- ja reikätietojen algoritmiavusteista mallintamista ei tutkittu case-tutkimuksessa, joten tämän enempää niiden mallintamiseen ei voida ottaa kantaa.

Rei'itysten, varausten ja aukotusten jälkeen jatketaan rakennusosien detaljointia. Kuten case-tutkimuksessa todetaan, algoritmiavusteisessa detaljoinnissa on tällä hetkellä vielä useita ongelmia. Case-tutkimuksessa todetaan myös, että väliseinien algoritmiavusteinen detaljointi on jossakin määrin mahdollista, joten muillakin rakennusosilla detaljointi voi olla toteutettavissa. Detaljoinnin jälkeen suunnitellut elementit on tarkastettava. Kuten jo aikaisemmin tutkimuksessa on todettu, tässä vaiheessa toteutettava tarkastus on hyvin tärkeää, koska ennen kuvatuotantoa muokkausten ja korjausten tekeminen on helpompaa. Jos virheitä huomataan, voidaan palata prosessissa takaisin päin ongelman vaatiman määrän. Kun ongelmat on korjattu ja korjatut elementit tarkastettu uudestaan, luodaan elementeistä valmistuskuvat. Valmistuskuvat tarkastavat sekä suunnittelija että vastaava rakennesuunnittelija.

6. CASE-TUTKIMUKSEN TULOSTEN ANALYSOINTI

Tulosten analysoinnissa kerrataan ja analysoidaan case-tutkimuksessa havaittuja ongelmia ja hyötyjä. Analysoinnissa pelkän toteamisen sijasta yritetään käsitellä asioita useasta näkökulmasta ja löytämään syitä ilmiöiden taustalla ja ratkaista ne. Näihin tietoihin nojaten yritetään löytää keinoja, joilla mahdollistettaisiin algoritmiavusteisen suunnittelun toimivuus elementtisuunnittelussa.

Ensimmäisessä osiossa kerrataan ja analysoidaan case-tutkimuksessa havaittuja ongelmia. Havaituista ongelmista pyritään löytämään sekä syy että ratkaisu. Toisessa osiossa tutkitaan, voidaanko algoritmiavusteista suunnittelua hyödyntää elementtisuunnittelussa. Viimeisessä osiossa selvitetään minkälaisia mahdollisuuksia ja hyötyjä algoritmiavusteisesta suunnittelusta voidaan saada.

6.1 Algoritmiavusteisen suunnittelun ongelmat

Käsitellään case-tutkimuksen tulokset tutkimuksen mukaisessa järjestyksessä. Case-tutkimuksessa havaitut ongelmat on jaettu kolmeen eri osioon: algoritmia ja mallintamista edeltäviin ongelmiin, algoritmiin ja mallintamiseen liittyviin ongelmiin ja Grasshopperin ja Teklan välisestä linkistä johtuviin ongelmiin. Algoritmia ja mallintamista edeltävät ongelmat -osiossa käsitellään lähtötietoihin ja IFC-mallin käsittelyyn liittyviä ongelmia. Kahdessa jälkimmäisessä osiossa tutkitaan algoritmia, mallintamista, ohjelmistojen välisiä haasteita ja koko prosessiin liittyviä ongelmia.

6.1.1 Algoritmia ja mallintamista edeltävät ongelmat

Case-tutkimuksen aluksi koottiin ja tarkastettiin lähtötiedot. Tarkastus koostui lähinnä tarvittavien lähtötietojen olemassaolon varmistuksesta. Tässä tai seuraavassa vaiheessa tulisi tehdä myös arkkitehdin IFC-mallin tarkastus sekä Solibrilla että visuaalisesti. Solibrilla kyettäisiin tehokkaasti tarkastamaan perustavanlaatuiset ongelmat ja visuaalisesti mahdollisesti isompia mallinnusvirheitä. Kuten case-tutkimuksessa todettiin, SimpleBimin visuaalisuus ei ole hyvä, joten visuaalisen tarkastuksen toteuttaminen on melko hankalaa. Kun väliseinät olivat mallinnettu, havaittiin arkkitehdin tehneen mallintamisessa virhe, jota ei SimpleBim visuaalisessa tarkastuksessa oltu havaittu.

Arkkitehdin mallintamisessa tekemää virhettä voidaan pitää ongelmallisena. Virhe on ongelmallinen kahdesta syystä: mallintamisvirhe on hankala havaita ja sen vaikutus

koko prosessin toimivuuteen kasvaa suunnittelun edetessä. Koska visuaalisella tarkastuksella tällaisen virheen havaitseminen on hankalaa, tulee virhe pahimmassa tapauksessa havaituksi liian myöhäisessä vaiheessa, jolloin pienestäkin ongelmasta voi muodostua iso ongelma. Case-tutkimuksessakin ongelma havaittiin liian myöhäisessä vaiheessa. Yksiselitteistä ratkaisua tähän ei ole. Jos mallinnusvirhe havaitaan liian myöhäisessä vaiheessa, ei ole järkevää palauttaa lähtötietona ollutta arkkitehdin IFC-tiedostoa, vaan virhe kannattaa korjata Teklassa. Jos virhe korjataan Teklassa, linkki algoritmin ja Teklan välillä on katkaistava, jotta algoritmi ei ylikirjoita käsin tehtyä muutosta.

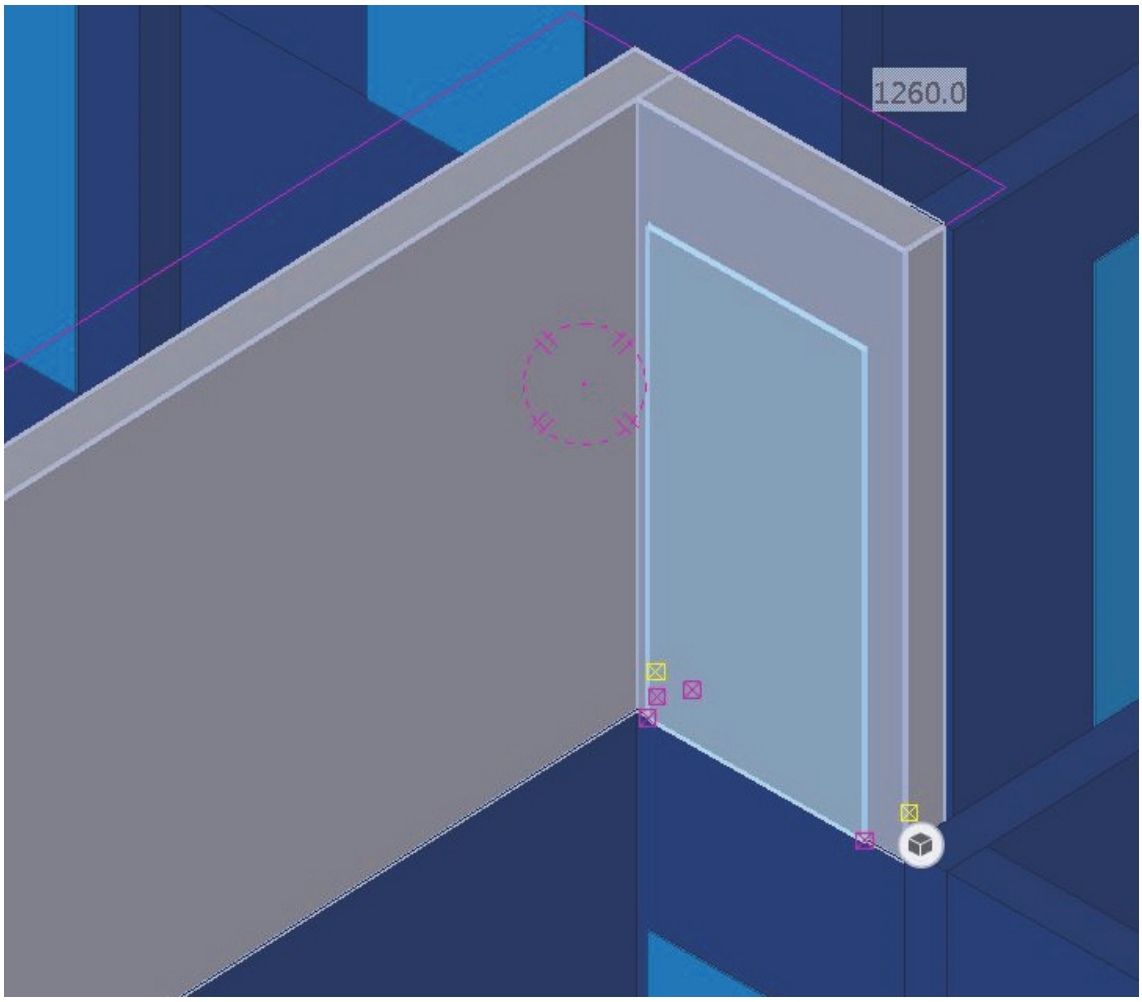
Lähtötietojen koonnin ja tarkastamisen jälkeen tehtiin väliseinien erottelu IFC-mallista. Tässä osiossa eroteltiin arkkitehdin IFC-mallista kantavat betoniset väliseinät muusta mallista käyttäen SimpleBim ohjelmistoa. Ensimmäinen ongelma havaittiin seinien erottelussa. Arkkitehdin tulisi määrittää kaikille osille load bearing -ominaisuus, mutta tätä ei oltu määritetty tai se oli määritetty väliseinille väärin. Tarkoituksena oli käyttää load bearing ja väliseinä ominaisuuksia filteröintiin, mutta tätä ei voitu käyttää.

Samanlaisia tiedon määrittämisen ongelmia ei tulisi esiintyä, sillä tietomallintaminen ja algoritmiavusteinen mallintaminen perustuvat eksaktisti määritettyyn tietoon. Algoritmi ja komponentit sekä Teklassa että Grasshopperissa eivät toimi, jos tieto on asetettu väärin. Samalla tavalla arkkitehdin virheestä voi aiheutua ongelmia muille suunnitteluosapuolille. Solibrilla tehtävä tarkastus voisi ratkaista nämä tietosisältöön liittyvät ongelmat. Tarkastukseen käytettävä aika todennäköisesti säästyy monin kertaisena, jos verrataan tilanteeseen, ettei tarkastusta tehdä ja myöhemmässä vaiheessa projektia havaitaan tietosisällöllinen virhe.

6.1.2 Algoritmiin ja mallintamiseen liittyvät ongelmat

Kun seinät oli eroteltu omaan IFC-tiedostoon, ne siirrettiin GeometryGymin avulla Rhinoon. Alkuun tehtiin virhe mallinnuslinjan valinnassa, kun valittiin seinän reunalinja mallinnuslinjaksi. Tämä johtui siitä, että se oli välittömästi saatavilla ja hyödynnettävissä. Väärästä mallinnuslinjasta ja arkkitehdin mallinnussuunnan vaihtelusta aiheutui ongelma siitä, että jotkin seinät mallintuivat väärin päin. Algoritmi kuitenkin korjattiin ja hyödynnettiin normaalistikin käytettävää seinän keskilinjaa mallinnuslinjana ja tällöin seinät saatiin mallinnettua oikeille sijainneille.

Tässä vaiheessa yritettiin tehdä seinien elementointia, mutta sitä ei saatu tehtyä. Case-tutkimuksen yhteydessä tehtiin listaus asioista, jotka vaikuttavat seinien elementointiin ja todettiin, että sellainen algoritmi, joka kykenisi ottamaan listan kaikki asiat huomioon, on mahdotonta muodostaa. Myöhemmin huomattiin myös, että osa seinistä ei ole toteutuskelpoisia, kuten kuvassa 44 esitetään. Väliseinän oviaukon toiseen pieleen on jäänyt pieni kaistale, jota ei ole mahdollista valmistaa.



Kuva 44. *Toteuttamiskelvoton väliseinä, jonka nurkkaliitos tulisi kääntää toisin päin.*

Kuvan 44 osoittamassa tilanteessa nurkkaliitos tulisi kääntää toisin päin. Käsin mallintamalla se onnistuisi helposti. Työn puitteissa ei kuitenkaan tutkittu, kyettäisiinkö tilanteeseen luomaan algoritmi, joka muuttaisi nurkan suunnan. Todettakoon myös, että mahdollisuus ongelmaan väliseinien välisissä nurkissa tiedostettiin jo ennen case-tutkimusta.

Herätetään tässä vaiheessa ajatus siitä, minkälainen hyöty ja arvo voidaan saada algoritmiavusteisesta elementtisuunnittelusta, jos elementointia ei voida tehdä algoritmiavusteisesti. Elementtijaon suunnittelu on kuitenkin yksi päätehtävä, jotta väliseinäelementit voidaan suunnitella ja detaljoida loppuun. Elementoinnin haastavuudesta huolimatta ei kuitenkaan täysin tyrmätä algoritmiavusteisen suunnittelun hyödyntämisestä elementtisuunnittelussa, vaan jatketaan analysointia.

Haasteena oli myös päättää, mallinnetaanko seinät kerros kerrallaan vai kaikki seinät yhdellä kertaa. Molempia keinoja kokeiltiin ja päätettiin mallintaa kaikki seinät kerrallaan. Tämä keino toimi case-tutkimuksen tapauksessa, mutta on huomioitava todelliset projektit, joissa suunnittelu tehdään lohko ja kerros kerrallaan. Tällöin olisi järkevää,

että myös algoritmi mahdollistaisi elementtien suunnittelun kerros kerrallaan. Tällä hetkellä algoritmi mallintaa kaikki seinät ja kaikki detaljoinnit yhdellä kertaa. Tämän lisäksi rakenneteknisestä näkökulmasta ei ole järkevää, että seinien raudoitus on jokaisessa kerroksessa sama, sillä monessa tapauksessa raudoitus tai muut liitososat kuten seinäkengät vaihtelevat kerroksittain. Kerrostieto lisätään yleensä start numberilla, jolloin ensimmäisen kerroksen seinillä start number on 1001 ja seuraavalla 2001 ja niin edelleen. Tätä ei kyetty toteuttamaan työn puitteissa, mutta se voisi olla mahdollista toteuttaa algoritmisesti, varsinkin jos seinät mallinnettaisiin algoritmilla kerroksittain.

Kuten on todettu, tietyt ohjelmistot ovat laajassa käytössä tällä hetkellä. Tutkimuksessa havaituista ongelmista osa liittyy ohjelmistoihin ja niiden toiminnallisuuksiin. Tulevaisuudessa ohjelmistot voivat kuitenkin vaihtua tai niihin tulla muutoksia, jolloin ongelmat voivat ratketa. Tällaisia skenaarioita on kuitenkin mahdotonta ennustaa, joten tärkeämpää olisi kehittää toimintamalleja niin, että ne ovat mahdollisimman vähän ohjelmistosidonnaisia. Tämän tutkimuksen keskeisenä tarkoituksena on tutkia algoritmiavusteisen suunnittelun soveltuvuutta elementtisuunnitteluun ja vasta toissijaisena tarkoituksena on tutkia tämän hetken ohjelmistojen soveltuvuutta algoritmiavusteiseen suunnitteluun.

Toistetaan lyhyesti detaljoinnissa havaittuja ongelmia. Oviaukkojen mallintamisessa oli ongelmina part cut -komponentin käyttämä aika ja jumiutuminen. Oviaukkojen mallintamisen algoritmissa olisi kehitettävää, sillä se saatiin tässä tapauksessa toimimaan vain siten, että algoritmi kokeilee jokaista oviaukkoa jokaiseen seinään. Tällä tavalla toteutettu algoritmi on hyvin epäkäytännöllinen. Se oli kuitenkin tämän tutkimuksen tapauksessa ja aikataulussa löydetty toimiva algoritmi, jolla aukot saatiin luotua oikein. Algoritmiin liittyy selvä jatkokehitystarve. Monimutkaisten komponenttien kuten vaaka- ja pystyliitos -komponenttien käyttö ei onnistu. Niiden käyttö voi olla täysin mahdotonta toteuttaa algoritmisesti, koska niihin liittyy sekä suunnittelijan tekemiä päätöksiä että komponentilla tehtäviin valintoihin liittyviä ongelmia, joita on todella vaikea ratkaista.

Komponentit, jotka voidaan toteuttaa Teklassa yhdellä hiiren painalluksella, on helppo toteuttaa algoritmiavusteisesti. Ainoa niiden toiminnassa havaittu ongelma on, että ne eivät joka kerta ladanneet oikeita esiasetuksia. Tämä kuitenkin korjaantui kun malli luotiin pakotetun sulkemisen jälkeen alusta alkaen uudelleen. Tämän jälkeen komponentti toimi oikein.

6.1.3 Grasshopperin ja Teklan välisestä linkistä aiheutuvat ongelmat

Tutkimuksen kannalta iso ongelma havaittiin, kun täysin detaljoitu Tekla-malli sekä Rhino ja Grasshopper -ohjelmistot avattiin uudelleen. Avattaessa Grasshopper pyrkii suorittamaan algoritmin uudelleen. Ohjelmisto ei kuitenkaan saanut vietyä algoritmia loppuun ja ohjelmat jouduttiin sulkemaan pakottamalla, kun algoritmin suoritus oli jat-

kunut kahden ja puolen tunnin ajan. Ongelmaa tutkittiin ja siihen löydettiin kaksi mahdollista syytä. Ensimmäinen mahdollinen syy on, että ongelma oviaukkojen algoritmissa, mikä aiheuttaa sen, että algoritmi pyrkii kokeilemaan jokaista aukkoa jokaiseen seinään. Toinen mahdollisuus on, että joka kerta kun algoritmi pyrkii mallintamaan väliseinään aukon, Tekla suorittaa uudelleen seinään liittyvät komponentit.

On mahdollista, että edellä mainituista mahdollisuuksista muodostuu yhdessä ketju, jossa aina kun oviaukkoa kokeillaan johonkin seinään, Tekla pyrkii suorittamaan siihen seinään liittyvät komponentit uudestaan. Syntyvä ketju on pitkä ja sen kestoa on mahdoton arvioida. Algoritmi saatiin toimimaan kytkemällä Grasshopperissa oviaukko- ja muut detaljointikomponentit pois päältä, kun seinät oltiin kertaalleen algoritmiavusteisesti detaljoitu Teklaan. Tämä ratkaisu edellyttää, että jos seiniin tulee jotakin muutoksia, tulee seinät ja kaikki niiden komponentit ensin poistaa, minkä jälkeen algoritmi voidaan suorittaa uudelleen. Muuten algoritmi ja Tekla ajautuvat uudestaan edellä esitettyyn ketjuun. Seinien poistamista ei kuitenkaan haluta tai voida tehdä, sillä kappaleiden poistaminen mallista aiheuttaa numeroinnin sekoittumisen. Numerointien sekoittuminen on vaarallista varsinkin, jos seinistä on tehty jo elementtikuvia, sillä kaikkien seinien yksilölliset numerot ja juoksevat numeroinnit sekoittuisivat toimenpiteen vuoksi. Vaikka uusi ja vanha seinä olisivat identtiset, ei niiden numerointeja voitaisi enää yhdistää toisiinsa.

Kuvatuotannossa havaittiin toinenkin ongelma. Tämä ongelma liittyy Grasshopperin ominaisuuteen, jossa algoritmitiedoston avaaminen suorittaa myös sen. Kun elementtikuva on tehty, ja Grasshopper avataan uudelleen, Grasshopper pyrkii suorittamaan koko luodun algoritmin uudestaan. Tällöin se myös pyrkii uudelleen mallintamaan seinät ja detaljoimaan ne, joka johti edellä todettuun ketjuuntumiseen ja ohjelmistojen pakotettuun sulkemiseen. Simuloitiin kuitenkin tilanne, jossa ongelmaa ei tulisi ja algoritmi suorittaisi itsensä loppuun, jolloin myös Teklan komponentit mallintuvat uudelleen. Tämä johti siihen, että Tekla tunnistaa muutokset seinissä, vaikka todellisuudessa niissä ei olisi tapahtunut muutosta. Koska Tekla havaitsee seinissä muutoksia, on kyseisen numerointisarjan elementit numeroitava uudelleen ja elementtikuvat päivitettävä. Ongelman välttämiseksi elementtikuvat voidaan jäädyttää, jolloin niihin ei tapahdu mitään muutoksia. Kuvien jäädyttämisessä on kuitenkin ongelma, koska silloin myöskään halutut muutokset eivät näy kuvaluettelossa tai kuvissa. Tällöin on siis mahdollista tehdä malliin muutoksia, mutta suunnittelijan tulee tietää hyvin tarkkaan mihin muutokset vaikuttavat, koska Tekla ei enää ilmoita muutoksista kuvaluettelossa.

Yhtenä osana ratkaisua tähän ja moneen edellä esitettyyn ongelmaan voisi olla kaikkien seinien mallintaminen kerroksittain. Seinien kerroksittain mallintamisen avulla kerrostieto, eli start number saataisiin kerroksittain algoritmilla lisättyä. Lisäksi raudoitukset ja muu detaljointi voitaisiin myös tehdä kerroksittain. Kerroksittain suunnittelu mahdollistaa prosessin hyödyntämisen todellisissa elementtisuunnitteluprojekteissa, joissa

alimman ja ylimmän kerroksen suunnittelun ajankohdalla on useiden kuukausien mittainen ero.

Jos algoritmi luotaisiin kerroksittain, voitaisiin algoritmin ja Teklan välinen linkki myös katkaista kerroksittain. Kun alimman kerroksen seinät saataisiin detaljoitua algoritmilla, tulisi algoritmin ja Teklan välinen yhteys katkaista, jotta vältyttäisiin jatkuvan kierteen ongelmalta ja mahdollistettaisiin kuvatuotannon toteuttaminen. Jos kaikki seinät olisivat mallinnettu yhdellä algoritmilla, jouduttaisiin algoritmin ja Teklan välinen linkki muiden kuin alimman kerroksen näkökulmasta katkaista liian aikaisin. Tällöin mitään mahdollisia muutoksia tai detaljointeja ylimmille kerroksille ei voitaisi tehdä algoritmisesti. Yhtenä isoimpana etuna algoritmiavusteisessa suunnittelussa pidetään muutoksiin reagointia. Kerroksittain mallintaminen mahdollistaisi muutoksiin reagoinnin paremmin kuin yhdellä kerralla kaikkien kerrosten mallintaminen. Tätä kerroksittain mallintamista ei kuitenkaan tutkimuksen aikataulusta johtuen kyetty tutkimaan.

Tutkimuksessa havaittiin, että algoritmi ajaa Teklassa tehdyn muutoksen yli. Kerroksittain mallintamalla ongelma voitaisiin välttää. Esimerkkinä tilanne, jossa algoritmilla on mallinnettu tietynlaiset raudoitukset Teklaan. Jos Teklassa joudutaan tekemään muutoksia raudoituksiin, Grasshopperin avaamisen jälkeen algoritmi korjaa Teklassa tehdyt muutokset algoritmin mukaiseksi. Kyseessä on sama Grasshopperin ominaisuus, josta on mainittu aikaisemmin tässä osiossa. Algoritmin ylikirjoittamisen ominaisuus on myös laadunvarmistuksellisesta näkökulmasta epätoivottu tilanne. Suunnittelija saattaa unohtaa purkaa Teklan ja algoritmin välisen linkin, jolloin todennäköisesti myöskään algoritmin tekemää muutosta ei huomata. Kerroksittain mallintamalla ja linkin purkamisella ylikirjoittamisen ongelma voidaan välttää.

6.2 Algoritmiavusteisen suunnittelun hyödynnettävyys ja toimivuuden edellytykset betoniväliseinäelementtien suunnittelussa

Tutkimuksen tavoitteiden mukaisesti tässä osiossa käsitellään millä tavalla algoritmiavusteista suunnittelua voidaan hyödyntää elementtisuunnittelussa. Osio käsittää kaksi näkökulmaa: mitä osa-alueita voidaan hyödyntää ja millä edellytyksillä mahdollistetaan algoritmiavusteisen suunnittelun hyödynnettävyys elementtisuunnittelussa.

6.2.1 Algoritmiavusteisen suunnittelun hyödynnettävyys elementtisuunnittelussa

Case-tutkimuksen tulosten perusteella ei voida tehdä yksiselitteistä johtopäätöstä algoritmiavusteisen suunnittelun hyödynnettävyydestä elementtisuunnittelussa. Tämä johtuu siitä, että case-tutkimuksen tulosten perusteella algoritmiavusteinen suunnitteluprosessi ei sovellu suoraan hyödynnettäväksi, mutta prosessia kehittämällä tai siinä havaittuja

ongelmia ratkaisemalla mahdollisesti kyettäisiin hyödyntämään uutta suunnitteluprosessia.

Diplomityön alussa esitettiin ajatus siitä, että mallintava suunnittelu tai algoritmiavusteinen suunnittelu eivät ole itseisarvoisia, vaan ne ovat työkaluja, joiden avulla on mahdollista löytää uusia keinoja toteuttaa suunnittelua ja tehostaa prosesseja. Tehostaminen voidaan saavuttaa usealla tavalla, kuten aikataulusäästöinä, rahallisena säästönä, suunnitelmien laadun parantumisena, virheiden löytämisenä tai eri vaihtoehtojen vertailuna. Case-tutkimuksen perusteella todetaan, että tasapainottelu algoritmin luomisen, kehittämisen ja algoritmista saatavan hyödyn välillä on hankalaa. Algoritmin luomiseen ja muokkaamiseen käytettävä aika tulee suhteuttaa siitä saatavaan hyötyyn, varsinkin, jos algoritmilla pyritään tekemään jotakin, johon on jo olemassa tehokas keino perinteisessä prosessissa. Lisäksi algoritmin luomiseen ryhtyessä on otettava huomioon, onko algoritmilla uudelleenkäyttömahdollisuutta. Edellä esitetyt asiat ovat niin laajoja ja monimutkaisia, että vastauksen löytäminen niihin yhden case-tutkimuksen avulla on käytännössä mahdotonta.

Case-tutkimuksessa todettiin useita ongelmia Grasshopperin ja Teklan välisen linkin toimivuudessa väliseinien detaljoinnissa. Havaittiin myös, että tutkimuksessa luotu algoritmiavusteinen elementtisuunnitteluprosessi ei tällä hetkellä ole täysin toimiva. Nämä ongelmat eivät kuitenkaan tarkoita, ettei algoritmiavusteista suunnittelua voitaisi hyödyntää elementtisuunnittelussa. Vaikka detaljointi ja kuvatuotanto aiheuttivat haasteita elementtisuunnittelussa, väliseinien mallintamiseen algoritmi toimi hyvin. Tutkimuksen perusteella todetaan algoritmiavusteiselle elementtisuunnittelulle kaksi mahdollista kehityssuuntaa: tehokas ja eksakti rakenteiden mallintaminen tai rajatun kokonaisuuden kokonaisvaltainen elementtisuunnittelu. Case-tutkimuksessa luodulla algoritmilla kyetään mallintamaan betoniset väliseinät samalla tavalla kuin ne on määritetty IFC-mallissa. Tästä algoritmista voitaisiin mahdollisesti hyödyntää tiettyjä moduuleita ja luoda muidenkin rakennusosien mallintamiseen omat algoritminsa. Eri rakennusosien algoritmeilla mahdollistettaisiin tilanne, jossa rakennusosat eroteltaisiin arkkitehdin IFC-mallista, minkä jälkeen ne tuotaisiin Rhinoon. Rhinoon tuomisen jälkeen Grasshopperilla kyettäisiin rakennusosille luotuja algoritmeja hyödyntämällä mallintamaan tehokkaasti kaikki eri rakennusosat Teklaan.

Toinen selkeä mahdollisuus algoritmiavusteisen suunnittelun hyödyntämiselle olisi jokin hyvin rajattu kokonaisuus, jossa on paljon toistoa ja elementtijako olisi yksinkertainen. Rajattu ja selkeä kokonaisuus voisi olla esimerkiksi porras- tai hissikuilu. Porras- ja hissikuilun tapauksessa vältyttäisiin elementtijaon algoritmiavusteisen suunnittelun ongelmilta, saataisiin hyödynnettyä algoritmiavusteisen suunnittelun tehokas ja eksakti mallintaminen sekä detaljointi. Rajatun kokonaisuuden -mahdollisuuteen liittyy olennaisesti myös se, että aikaisemmin esitetyt ongelmat kerroksittain mallintamisessa, algoritmin ja Teklan välisessä linkissä ja tietyissä detaljointikomponenteissa ratkaistaan.

6.2.2 Algoritmiavusteisen suunnittelun hyödynnettävyyden edellytykset

Algoritmiavusteiseen detaljointiin liittyy ongelmia, mutta niistä huolimatta detaljoinnin toteuttaminen algoritmiavusteisesti voisi olla mahdollista. Algoritmiavusteisen detaljoinnin mahdollistamisessa tulisi huomioida muutamia asioita algoritmin luomisessa ja algoritmin ja Teklan välisessä linkissä. Kuten jo edellisessä osiossa todettiin, rakennusosat tulisi mallintaa kerroksittain. Tällöin myös detaljointi tehtäisiin kerroksittain. Kerroksittaiseen suunnitteluun on kaksi olennaista syytä. Ensimmäinen syy on ohjelmistotekniset ongelmat, jotka on esitelty edellisessä osassa. Toinen syy muodostuu siitä, että yleensä alimmissa kerroksissa kuormat ovat suurimmat, jolloin liitos-osat ja raudoitukset tulee olla vahvemmat. Lisäksi, kuten jo aikaisemmin todettiin, alimman ja ylimmän kerroksen suunnittelun välillä on useiden kuukausien ero, joten myös algoritmiavusteinen suunnittelu tulisi tehdä kerroksittain.

Detaljoinnin toimivuuden edellytyksenä on myös ymmärtää Teklan ja Grasshopperin välisestä linkistä johtuvat ongelmat. Kun Grasshopper avataan, Grasshopper pyrkii suorittamaan algoritmin. Tämä aiheutti ongelman algoritmin oviaukkojen mallintamisessa. Teklan ja Grasshopperin väliseen linkkiin liittyy olennaisesti myös se, milloin linkki tulisi katkaista. Siinä vaiheessa, kun seinät halutaan aukottaa ja detaljoida, on varmistuttava siitä, että laajamittaisia muutoksia niihin ei enää tule, sillä tässä vaiheessa myös linkki Teklan ja Grasshopperin välillä on katkaistava. Linkin katkaisemisen jälkeen muutosten tekeminen algoritmilla on mahdotonta, mutta muutoksia voidaan tehdä perinteisin menetelmin Teklalla. Linkin katkaisemisella on tarkoitus taata, että algoritmi ei aiheuta detaljointi ja kuvatuotanto -vaiheissa ongelmia. Jos detaljointiin ja jatkuvaan kierteeseen liittyvät ongelmat saadaan tulevaisuudessa ratkaistua, on mahdollista säilyttää linkki niin kauan, kunnes detaljoidun kerroksen seinistä aletaan tehdä valmistuskuvia. Tällöin linkki katkaistaan siitä syystä, että joka kerta kun Grasshopper avataan, Grasshopper suorittaa algoritmin, joka aiheuttaa Teklassa näennäisiä muutoksia. Vaikka muutokset eivät ole todellisia, Tekla ymmärtää ne todellisiksi muutoksiksi ja vaatii kaikkien numerointisarjojen uudelleennumeroinnin ja kyseisten kuvien päivittämisen, jotka olivat osa algoritmia. Huomion arvoinen seikka on, että nykyisessä mallintavassa suunnitteluprosessissa on mahdollista samaan aikaan sekä detaljoida että tehdä valmistuskuvia saman numerointisarjan elementeistä.

Kuten case-tutkimuksessa havaittiin, kaikkia vaikeimpia liitoskomponentteja ei voida algoritmiavusteisesti toteuttaa. Vaikka nämä liitokset tehtäisiin Teklassa mallintamalla, algoritmi ei ylikirjoita algoritmiavusteisesti tehtyjä liitoksia. Algoritmi ylikirjoittaa ainoastaan muutoksia jotka koskevat algoritmilla tehtyjä komponentteja. Teklalla kyetään myös tekemään elementtijako jo täysin detaljoiduille väliseinille. Tekla katkaisee seinän ja siirtää kaikki komponentit toiselle seinälle. Teklassa on myös kopiointiominaisuus, jolla voidaan kopioida kaikki valittuun seinään liittyvät komponentit toiseen seinään.

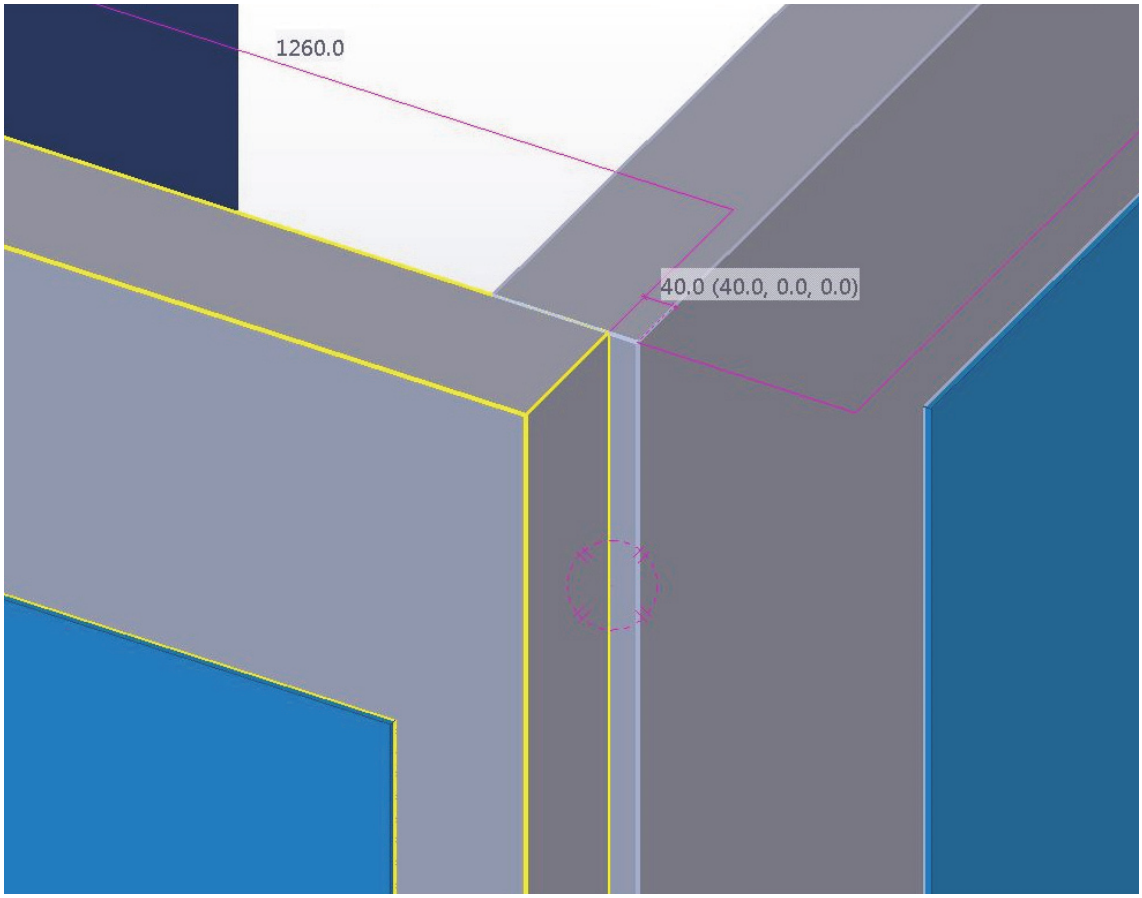
Todettakoon, että edellä mainitut ongelmat eivät suoranaisesti estä osittaista algoritmiavusteista detaljointia. Suunnittelijan on kuitenkin ymmärrettävä tarkkaan näiden edellä mainittujen asioiden vaikutukset suunnitteluprosessiin, jotta suunnitteluprosessin toimivuudelle on riittävät edellytykset.

Vaikka detaljointi voitaisiin periaatteessa tehdä algoritmiavusteisesti, siihen liittyy vielä muutama hankala ongelma. Jos algoritmi muokataan kerros kerrallaan mallintavaksi ja detaljoivaksi, seuraava ongelma kohdataan, kun kerrokset sisältävät seiniä, joihin on suunniteltu erilaisia raudoituksia tai muita detaljointeja. Tällöin algoritmilla jouduttaisiin mallintamaan kaikkiin tietyn kerroksen seiniin samanlaiset raudoitukset ja muut detaljoinnit. Virheellisen tiedon mallintaminen tietomalliin on väärin. Virheellinen tieto aiheuttaa todella vakavan laadunvarmistuksellisen ongelman. Lähtökohtaisesti haastava seinä tai jokin muu rakennusosa suunnitellaan huolella erikseen, niin että se suunnitellaan yhdellä kertaa oikein. Millään keinolla ei voida varmistua, jos jokin elementti detaljoidaan algoritmilla väärin, että se muistettaisiin jälkikäteen erikseen suunnitella. Tästä kerroksen sisäisestä detaljointiin liittyvästä ongelmasta johtuen tulisi tutkia lisää, voitaisiinko seiniä tai muita rakennusosia jakaa algoritmissa yhä rajatumpiin osioihin, mikä mahdollista kerroksen sisäisesti erilaisten detaljointien algoritmiavusteisen suunnittelun.

Trimble on myös kehittänyt elementtisuunnittelun työkaluja uusimmissa päivityksissään Teklassa. Uusi detailing manager -työkalu on tarkoitettu elementtirakenteiden suunnitteluun ja detaljointiin. Työkaluun syötetään halutut komponentit ja niiden esiasetukset, valitaan halutut rakenteet ja komponentti suorittaa kaikki valitut komponentit oikeilla asetuksilla. Tämä on pääpiirteittäin idealtaan sama kuin algoritmiavusteinen detaljointi. Työkalulla etuna on kuitenkin se, että sillä voidaan helpommin valita rakenteet, joihin detaljointi halutaan tehdä. Tällöin esimerkiksi kaikkiin seiniin ei tarvitse luoda samoja detaljointeja, jolloin jossakin seinissä on väärät detaljoinnit. Koska algoritmiavusteisesti ei tällä hetkellä kyetä yhtä tehokkaasti tekemään detaljointia kuin detailing manager -työkalulla, tulee arvioida, onko algoritmiavusteista detaljointia hyödyllistä pyrkiä kehittämään. Yksiselitteistä vastausta tähän ei tutkimuksen avulla saatu, vaan asia vaatisi lisätutkimusta.

Tähän mennessä esitellyt asiat ovat liittyneet tutkimuksessa käytettyihin ohjelmistoihin. Seuraavaksi käsitellään enemmän yleisellä tasolla algoritmiavusteisen suunnittelun hyödyntämisen edellytyksiä. Mallintaville osapuolille, varsinkin arkkitehdille on asetettava vaatimuksia, jotta algoritmiavusteinen suunnitteluprosessilla on tosiasialliset mahdollisuudet toimia halutulla tavalla. Tärkeimpänä vaatimuksena on mallinnustarkkuuden parantaminen ja virheiden vähentäminen. Kuvassa 45 esitetään arkkitehdin mallinnustarkkuudesta johtuva virhe Teklassa. Tätäkään virhettä ei voitu havaita ennen kuin Teklassa, jossa visuaalisuus on parempi. Tässä tilanteessa mallintamisvirheestä ei olisi tullut isompaa ongelmaa, sillä liitoskomponentti tämän tyyppisessä nurkkaliitoksessa korjaa virheen. Tästä huolimatta arkkitehdin ja muiden mallintavien osapuolten mallinnustarkkuuden tulee olla paljon parempi, sillä pahimmassa tapauksessa näin huono mal-

linnustarkkuus ja niiden aiheuttamien ongelmien myöhäinen havaitseminen voivat aiheuttaa merkittäviä ongelmia. Muiden osapuolten tulee pystyä luottamaan saatavien lähtötietojen tarkkuuteen ja oikeellisuuteen, koska jokainen osapuoli on kuitenkin vastuussa omista suunnitelmista ja niiden oikeellisuudesta.



Kuva 45. *Arkkitehdin mallinnustarkkuudesta johtuva virhe.*

YTV2012 osa 3 arkkitehtisuunnittelu antaa vaatimuksia ja ohjeita mallin tarkkuustasoihin ja mallinnettaviin rakenteisiin eri hankevaiheesta riippuen. Seinille on toteutusvaiheessa määrätty käyttämään tarkkuustasoa 2. Tässä tasossa sijainti ja geometria on mallinnettu vaatimusten mukaisesti, rakennetyyppi määritelty ja oikean niminen ja tuoteosat mallinnettu niin, että kappalemäärät ja muu oleellinen määrätieto saadaan tuotetyypeittäin mallista (Yleiset Tietomallivaatimukset 2012). Edellisessä kappaleessa käsitelty mallinnustarkkuus tulisi olla tämän määräyksen mukainen. Määräys ei ole täysin yksiselitteinen, joten projekteissa on yleensä tarkennettava näitä vaatimuksia. Case-tutkimuksessa havaittiin, että load bearing -ominaisuutta ei oltu asetettu kantaville väliseinille oikein. Tämä tulisi tarkkuustason 2 mukaisesti olla asetettu kantaville väliseinille. Samaan tarkkuustason 2 vaatimukseen sijoittuu kuvan 45 mallinnustarkkuuden virhe.

Mallinnustarkkuuteen liittyy myös mallinnussuunta. Case-tutkimuksen alkuvaiheessa havaittiin mallinnussuunnista johtuva virhe, koska seinät mallinnettiin seinän alapään toisen reunalinjan mukaisesti. Mallinnuslinja vaihdettiin seinän alapään keskilinjaan,

jonka jälkeen mallinnussuunnasta johtuva ongelma korjaantui. Todettakoon, että vääräs-tä tai vaihtelevista mallinnussuunnista voi aiheutua ongelmia muissa rakennusosissa. Ongelmia voidaan mahdollisesti kohdata esimerkiksi betonisandwich-seinillä, jos mallinnussuunta on väärin päin. Sandwichien valmistuskuvissa kuvan suunta on niin, että alkupiste on vasemmalla ja loppupiste oikealla. Kun arkkitehti mallintaakin seinät väärin päin ja muuttaa vain seinän rakenteet toisin päin, seinä on valmistuskuvassa peilikuvana, eikä elementtiä voida tämän piirustuksen mukaisesti valmistaa.

Tähän ongelmaan nähdään kaksi mahdollista ratkaisua: arkkitehtien mallintamiskäytäntöihin on tultava muutos ja mallintamissuunta tulee toistua kaikissa rakenteissa samalla tavalla tai kehitetään algoritmi, joka huomioi väärän mallinnussuuntaa. Teoriassa yksinkertaisin ratkaisu olisi arkkitehdin mallintamiskäytäntöjen muuttaminen, mutta tutkijalla ei ole niin kattavaa tietämystä arkkitehtien mallintamisohjelmistoista tai -käytännöistä, että voitaisiin todeta tämän olevan helpoin tai käyttökelpoisin ratkaisu.

6.3 Algoritmiavusteisen suunnittelun mahdollisuudet

Edellisissä osioissa on koottu ja analysoitu erilaisia havaittuja ongelmia. Seuraavaksi käsitellään enemmänkin algoritmiavusteisesta suunnittelusta saatavia hyötyjä ja mahdollisuuksia.

Tutkimuksen tulosten mukaan saadaan hyötyä mallinnustarkkuuden parantumisesta. Tämä vaatii myös sen, että lähtötietona saatavissa malleissa mallinnustarkkuus kehittyy. Tutkija on itse kohdannut projektityössä, ettei useissa kohteissa arkkitehdin IFC-mallia päivitetä muutosten mukaan. Viralliset dokumentit ovat 2D-piirustuksia, jolloin arkkitehdilla ei ole välttämättä halua päivittää jokaista muutosta myös tietomalliin. Tämä taas on ongelma, johon voitaisiin jo projektin alkuvaiheessa puuttua, kun asetettaisiin vaatimukset tietomallin päivittämiselle. Tulevaisuudessa myös 2D-piirustukset voivat hävitä täysin tai ainakin menettää asemansa ainoana virallisena dokumenttina, jolloin myös tietomallien tarkkuus ja päivittäminen tulee kehittyä.

Algoritmiavusteinen suunnittelu parantaa mallinnustarkkuuden lisäksi mallintamisen nopeutta. Case-tutkimuksessa luotiin algoritmi ainoastaan väliseinille, mutta tulevaisuudessa voitaisiin päästä tilanteeseen, jossa kaikille rakennusosille olisi oma algoritmi, joiden avulla mallintaminen tehtäisiin. Tämä olisi myös pitkällä aikavälillä tavoiteltava tilanne, koska ilman näitä algoritmeja koko elementtisuunnitteluprosessin toimivuutta kokonaiseen rakennushankkeeseen on vaikeaa tutkia. Huomataan myös, että tässä tutkimuksessa on nyt tutkittu ja kehitetty prosessi, joka pyrkii esittämään miten algoritmiavusteinen suunnittelu toteutetaan betonirakenteisten väliseinien elementtisuunnittelussa. Työssä kehitettiin myös prosessi, joka esittää algoritmin luomisen prosessia. Näiden avulla pitäisi pystyä toistamaan prosessi muillekin rakennusosille. On huomioitava, että prosessit eivät ole täysin identtisiä eri rakennusosien välillä, mutta samankaltaisia vaiheita rakennusosien algoritmin luomiseen liittyy.

Riippumatta siitä, kuinka hyvin eri detaljointikomponentit tai vastaavat ohjelmistoista riippuvaiset ongelmat saadaan ratkaistua, varmaa on kuitenkin, että algoritmiavusteisesta suunnittelusta tullaan saamaan hyötyä elementtisuunnittelussa. Täysin yksiselitteisesti ei voida kuitenkaan tulkita tämän tutkimuksen perusteella mikä tai mitkä osiot algoritmiavusteisesta suunnittelusta tullaan hyödyntämään tulevaisuudessa. Yksi vaihtoehto voi olla, että algoritmiavusteisesti mallinnetaan arkkitehdin IFC-mallin perusteella koko rakennus ilman detaljointia. Toisaalta, jos ohjelmistojen välisen linkin ja detaljointikomponenttien ongelmat ratkaistaan, voidaan päästä tilanteeseen jossa joko osa tai kaikki elementtien detaljoinnista tehdään algoritmiavusteisesti.

Tulevaisuudessa tilanne elementtisuunnittelussa voi olla se, että tietomallit ovat virallisia suunnitteludokumentteja, eikä elementeistä tarvitse tehdä valmistuskuvia ollenkaan. Tällaisessa skenaariossa algoritmiavusteinen mallintaminen luo vielä enemmän arvoa, kuin se loisi perinteisessä elementtipiirustus-elementtisuunnittelussa.

7. JOHTOPÄÄTÖKSET

Diplomityön lopuksi arvioidaan tutkimuksen onnistumista, mahdollisia jatkotutkimusaiheita ja tehdään yhteenveto tutkimuksesta. Todettakoon jo tässä vaiheessa lyhyesti tutkimuksen onnistuneen kokonaisuutena ja tutkimuksesta saatiin tärkeitä tuloksia. Tutkimuksen aikana nousi esille monta jatkotutkimusaihetta, joita olisi alan kehittämisen kannalta tärkeä tutkia.

7.1 Tutkimuksen tavoitteiden saavuttaminen

Johdanto-kappaleessa on esitelty tutkimuksen tavoitteet. Ne jaettiin tiedollisiin ja teollisiin tavoitteisiin. Tarkastellaan jokaista tavoitetta ja sen saavuttamista yksitellen. Ensimmäisenä tavoitteena oli kuvata mallintavan elementtisuunnittelun prosessi. Tämän prosessin kuvaamiseen hyödynnettiin sekä tutkijan että Ramboll Finland Oy:n osaamista ja Ramboll Finland Oy:ltä saatua materiaalia. Elementtisuunnittelun toteutusvaiheen prosessi on hiottu hyvin pitkälle satojen projektien avulla. Prosessi edustaa yhden yrityksen tapaa tehdä mallintavaa suunnittelua.

Tutkimuksen toisena tavoitteena oli kuvata algoritmiavusteinen elementtisuunnitteluprosessi betonisille väliseinäelementeille. Prosessi luotiin mallintavan suunnitteluprosessin ja teoriaosuudesta saadun tiedon avulla. Case-tutkimuksessa testattiin suunnitteluprosessin toimivuutta. Tutkimuksen lopputulemana todettiin, että yksiselitteisesti ei voida todeta prosessin toimivuutta tai hylätä sitä. Todettiin myös, että luotua prosessia ei voida hyödyntää esitetyssä muodossa, sillä prosessi sisältää vielä liian paljon toteutuksen kannalta merkittäviä ongelmia. Jotta voitaisiin luoda algoritmiavusteisesta elementtisuunnittelusta toimiva suunnitteluprosessi, tulisi asiaa tutkia ja testata laajamittaisemmin oikeissa projekteissa. Tutkimuksesta saadun kokemuksen perusteella algoritmiavusteisen suunnitteluprosessin jatkokehitykselle ja hyödyntämiselle on olemassa kaksi potentiaalista kehityslinjaa. Algoritmiavusteista prosessia voidaan soveltaa pelkästään rakenteiden mallintamiseen tai, jos ohjelmistojen väliset ongelmat saadaan ratkaistua, prosessia voidaan hyödyntää koko suunnitteluprosessissa tai sen osakokonaisuutena. Nämä kaksi tapausta tulisi tarkemmin testata todellisissa projekteissa, jotta voitaisiin tehdä pidemmälle vietyjä johtopäätöksiä.

Case-tutkimuksesta saatiin tulokseksi myös, että algoritminen mallintaminen on nopeampaa ja tarkempaa perinteiseen mallintamiseen verrattuna. Elementtisuunnitteluprosessi ei suuressa mittakaavassa muuttunut merkittävästi, sillä samat piirteet ja osiot toistuvat myös algoritmiavusteisessa prosessissa.

Algoritmiavusteista suunnitteluprosessia tulisi kehittää niin, että suunnittelu voitaisiin tehdä kerroksittain, jolloin kerroksen komponentit olisi mahdollista kytkeä pois päältä, kun detaljointi on saatu kerroksen osalta valmiiksi. Valmiin kerroksen komponentit eivät enää sen jälkeen rasittaisi suunnitteluprosessia. Prosessi olisi tällöin myös lähempänä lähtökohtana ollutta mallintavaa suunnittelua. Kerroksittain kytketyt komponentit estävät myös kuvatuotannon numerointiin ja päivityksiin liittyvät turhat toimenpiteet. Lisäksi kuvatuotantoa varten komponentit on myös kyseisistä numerointisarjoista kytkettävä pois, ettei niiden numeroinnit mene uusiksi ja kuvia joutuu päivittämään näistä syistä. Tutkimukselle asetettu tavoite selvittää algoritmiavusteisen suunnittelun vaikutukset betoniväliseinäelementtien suunnitteluprosessiin saavutettiin ja tutkimus tuotti kattavasti tietoa sekä prosessin toteutuksesta että ohjelmistojen käyttöön liittyvistä rajoitteista.

Tutkimuksen teoriaosuudessa selvitettiin algoritmiavusteisen suunnittelun hyödyntämistä rakennusalalla. Kirjallisuustutkimuksen avulla selvitettiin algoritmiavusteisen suunnitteluprosessin perusteita sekä menetelmän hyödyntämistä Suomessa ja ulkomailla. Selvitys osoitti, että varsinkin ulkomailla tutkimusta ja monimutkaisia projekteja toteutetaan jatkuvasti. Algoritmiavusteista suunnittelua sovelletaan laajasti erilaisten rakenteiden, muotojen sekä julkisivujen optimointiin. Rakenteiden suunnittelussa ja optimoinnissa hyödynnetään myös FEM-laskentaa. Kirjallisuusselvityksen avulla saatiin selkeä ja kattava käsitys algoritmien hyödynnettävyydestä.

Työlle asetetut teolliset tavoitteet sisälsivät kaksi pääkohtaa: luoda betoniväliseinäelementin algoritmiavusteinen suunnitteluprosessikaavio sekä tuottaa uudelleen hyödynnettävä menetelmä tai prosessi elementtisuunnitteluun soveltuvan algoritmin luomiseen. Perustuen kirjallisuusselvityksestä saatuun tietoon prosessille luotiin liitteessä A esitetty prosessikaavio. Prosessikaaviota noudattaen tuotettiin väliseinäelementin mallintamiseen ja detaljointiin algoritmi, jota testattiin case-kohteen avulla. Vaikka prosessia ei saatu vielä kehitettyä kaikilta osin täysin toimivaksi, tuotti tutkimus uusia ratkaisuehdotuksia, joilla prosessia ja algoritmin toteutusta voidaan parantaa. Koska tässä työssä tuotettu algoritmi on kehitetty ja testattu betoniväliseinien suunnitteluun, ei tämän kokemuksen perusteella voida vielä kovin hyvin arvioida prosessin soveltuvuutta muiden rakennusosien suunnitteluun. Tuotettu algoritmi toimii kuitenkin hyvänä lähtökohtana, jota voidaan lähteä jatkotutkimuksissa muokkaamaan uusille rakennusosille.

7.2 Jatkotutkimusaiheet

Tutkimuksen ja tulosten analysoinnin aikana kävi selväksi, että tällä tutkimuksella tehtiin vasta pintaraapaisu algoritmiavusteisen suunnittelun hyödyntämiseen elementtisuunnittelussa. Seuraavassa on esitetty työn aikana esille tulleet keskeiset jatkotutkimustarpeet.

Case-tutkimuksen alussa ja lopussa havaittiin, että arkkitehdin mallintamisessa oli puutteita tai virheitä. Alussa havaitut puutteet liittyivät tietomallin tietosisältöön ja lopussa havaitut virheet arkkitehdin mallinnustarkkuuteen. Nämä puutteet ja virheet aiheuttavat projektin aikana useita ongelmia, joiden suuruus ja vaikuttavuus vaihtelevat. Tietomallin tietosisältöön liittyvä ongelma voitaisiin ratkaista Solibrilla tai vastaavalla ohjelmistolla, jolla tietomallin tietosisältö voidaan tarkastaa. Tietosisällön tarkastamiseen tulisi kehittää ohjelmistoille tarkempi säännöstö YTV2012 julkaisun pohjalta. Tarkemman säännöstön kehittäminen vaatisi ison otannan toteutettuja projekteja, joista tietoa voitaisiin kerätä ja säännöstö muodostaa. Mallinnustarkkuuteen liittyvien ongelmien ratkaiseminen on hieman hankalampaa. Mallin tarkastustoimenpiteitä on hyvin hankala määräänsä enempää kehittää, mutta mahdollisessa ratkaisussa vaadittaisiin arkkitehdeiltä parempaa mallinnustarkkuutta. Mallinnustarkkuuteen ja tietomallin tietosisältöön liittyvät virheet voivat johtua myös siitä, että jokin tai jotkin suunnitteluosapuolet eivät ymmärrä mihin asioihin pienikin virhe voi vaikuttaa, jolloin pieniin mallinnustarkkuuden tai tietosisällön virheisiin ei osata kiinnittää huomiota.

Tutkimuksen rajauksesta johtuen talotekniikan vaatimia varausten ja läpivientien algoritmiavusteista mallintamista ei tutkittu. Prosessin toimivuuden kannalta olisi tärkeää millä tavalla niiden talotekniikan varausten ja läpivientien algoritmiavusteinen mallintaminen onnistuu. Tällä hetkellä varausten ja läpivientien mallintaminen Teklaan onnistuu kohtalaisen tehokkaasti. Tulisikin tutkia, voisiko algoritmiavusteinen mallintaminen tehostaa talotekniikan varausten ja läpivientien mallintamista sekä selvittää miten niiden mallintaminen soveltuisi nykyiseen algoritmiavusteiseen prosessiin.

Yhtenä tärkeimpänä jatkotutkimusaiheena voidaan pitää elementtisuunnittelun kaikkien rakennusosien algoritmiavusteisen suunnittelun toimivuuden testausta. Tässä tutkimuksessa tutkittiin ainoastaan väliseinän soveltuvuutta algoritmiavusteiseen prosessiin. Jo yhden rakennusosan prosessissa havaittiin useita ongelmia, mutta myös paljon mahdollisuuksia. Jatkotutkimuksena tulisi selvittää voidaanko prosessi toistaa muillekin rakennusosille ja toistuvatko väliseinän algoritmiavusteisessa suunnittelussa havaitut ongelmat myös muiden rakennusosien prosesseissa. Jos samat ongelmat eivät toistu kaikilla rakennusosilla, tulisi selvittää voitaisiinko näiden rakennusosien suunnittelussa hyödyntää algoritmiavusteista elementtisuunnittelun prosessia.

Tutkimuksessa havaittiin, että algoritmiavusteisessa prosessissa lähtötiedot ja niiden sisältö ovat tärkeitä. Samalla kehittyi tarve lähtötietojen vaatimuslistaukselle, jossa eri suunnitteluosapuolille asetettaisiin algoritmiavusteisen prosessin toimivuuden kannalta olennaiset vaatimukset. Vaatimuslistaus soveltuukin hyvin jatkotutkimusaiheeksi. Tämän tutkimuksen perusteella toimivan vaatimuslistauksen luominen on mahdotonta, sillä sen kehittäminen vaatii tietoa useista erilaisista elementtisuunnitteluprojekteista, joiden avulla listaus luotaisiin. Kun tutkimuksen otanta olisi tarpeeksi suuri, olisi mahdollista löytää toistuvuuksia ja tiettyihin osa-alueisiin sidonnaisia vaatimuksia.

Olennaisena osana algoritmiavusteista suunnittelua oli algoritmin uudelleenkäyttö. Olisi hyödyllistä tutkia millä tavalla luotua algoritmia voitaisiin uudelleenkäyttää. Lisäksi tulisi tutkia, kuinka luotu algoritmi toteuttaa ryhmittelyyn, nimeämiseen ja jäsentelyyn liittyviä asioita, joita käsiteltiin teoria-osuudessa. Näitä voitaisiin tutkia esimerkiksi testiryhmän avulla. Jos tulevaisuudessa ollaan tilanteessa, jossa algoritmeja hyödynnetään useimmissa projekteissa, tulee algoritmit olla helposti omaksuttavissa ja muokattavissa. Algoritmi on nyt luotu moduuleihin, ja ajatuksena on, että näitä moduuleita yhdistelemällä ja kevyesti muokkaamalla voitaisiin luoda muihin rakennusosiin soveltuva algoritmi.

7.3 Yhteenveto

Tutkimuksessa selvitettiin millä tavalla nykyiseen mallintavaan elementtisuunnitteluprosessiin soveltuu algoritmiavusteiset menetelmät. Aluksi tutkittiin sekä algoritmiavusteisen suunnittelun että perinteisen mallintavan elementtisuunnittelun teoriaa ja taustoja. Sekä Suomessa että ulkomailla ilmeni useita erilaisia käyttökohteita algoritmiavusteisille suunnittelumenetelmille. Arkkitehdit hyödyntävät luonnon ja evoluution mekanismeja pyrkiessään löytämään erilaisia muotoja. Rakennesuunnittelijat hyödyntävät myös evolutiivisia menetelmiä rakenteiden optimoinnissa. Muun muassa teräsristikon muotoa, paarteiden määrää tai niiden profiileja voidaan optimoida hyödyntämällä erilaisia puoliautomaattisia optimointialgoritmeja. Optimoinnissa paras mahdollinen ratkaisu voidaan määritellä hinnan, kilomäärän, paarresauvojen lukumäärän tai monen muun reunaehdon mukaisesti.

Tutkimuksen teoriaosuuden avulla luotiin algoritmiavusteinen elementtisuunnitteluprosessi, jonka lähtökohtana oli perinteinen mallintava suunnitteluprosessi. Case-tutkimuksessa tutkittiin algoritmiavusteisen elementtisuunnitteluprosessin toimivuutta betonisten väliseinäelementtien suunnittelussa. Case-tutkimuksen kohteena oli todellinen elementtisuunnitteluprojekti. Tästä projektista saatiin case-tutkimukseen lähtötiedoksi asuinkerrostalon IFC-malli, josta eroteltiin tutkimusta varten kantavat betoniset väliseinät. Algoritmin luominen ja kehittäminen olivat monimutkaisia oppimisprosesseja. Väliseinäelementin suunnitteluprosessi jaettiin eri osa-alueisiin, jotta osa-alueiden algoritmeja voitiin kehittää erillään toisistaan. Case-tutkimuksen lopputuloksena todettiin, että algoritmiavusteinen suunnittelu ei tällä hetkellä ole prosessikaavion esittämällä tavalla toteutettavissa.

Tutkimuksen perusteella algoritmiavusteiselle elementtisuunnitteluprosessille löydettiin kaksi selkeää linjaa, joiden avulla algoritmiavusteinen suunnitteluprosessi voitaisiin kehittää toteutettavaksi. Ensimmäinen mahdollisuus olisi kehittää algoritmiavusteista prosessia niin, että hyödynnettäisiin ainoastaan algoritmiavusteista mallintamista. Tällöin arkkitehdin IFC-mallin perusteella rakennus mallinnettaisiin algoritmiavusteisesti Teklaan tehokkaasti ja tarkasti, jonka jälkeen detaljointi ja kuvatuotanto tehtäisiin Teklassa. Toisena mahdollisuutena olisi pyrkiä ratkaisemaan tutkimuksen aikana havai-

tut ongelmat ja hyödyntämään algoritmiavusteista prosessia kokonaisuutena tai sen osa-alueita.

Tutkimuksessa havaittujen ongelmien ja mahdollisuuksien lisäksi kyettiin tuottamaan runsaasti arvokasta tietoa algoritmiavusteisesta prosessista ja löydettiin selkeät linjat prosessin kehittämiselle. Suomessa on tehty kohtalaisen vähän tutkimusta algoritmiavusteisesta suunnittelusta, joten tällä tutkimuksella on myös oma arvonsa tässä kategoriassa. Loppupäätelmänä tutkimuksen perusteella voidaan todeta, että tämä tutkimus antaa hyvän lähtökohdan jatkaa algoritmiavusteisen suunnittelun tutkimusta sekä elementtisuunnittelun että rakennesuunnittelun osilta.

LÄHTEET

Davis, D. (2013). Modelled on Software Engineering: Flexible Parametric Models in the Practice of Architecture, RMIT University,

Eastman, C., Teicholz, P., Sacks Rafael & Liston Kathleen (2008). BIM handbook: a guide to building information modeling for owners, managers, designers, engineers, and contractors, Wiley, Hoboken, N.J, 490 p.

Elementtirakentamisen historia, Betoniteollisuus Ry, web page. Available (accessed 17.1.2017):
http://www.elementtisuunnittelu.fi/fi/valmisosarakentaminen/elementtirakentamisen-historia.

(2017). Elementtisuunnittelun prosessikaavio, Ramboll Finland Oy, Espoo, julkaisematon selvitys, 2 p.

Grasshopper-Tekla Live Link, Trimble Solutions Corporation, web page. Available (accessed 24.01.2017): https://teklastructures.support.tekla.com/not-version-specific/en/ext_grasshopperteklalink.

Harmanen, M. (2010). Betonielementtikohteiden tietomallipohjainen suunnitteluprosessi, Tampereen teknillinen yliopisto, Talouden ja rakentamisen tiedekunta, Tampere, 85 p.

Humppi, H. (2015). lgorithm-Aided Building Information Modeling:
Connecting Algorithm-Aided Design and Object-Oriented Design, Tampereen teknillinen yliopisto, Arkkitehtuurin laitos, Tampere, 176 p.

Hytönen, Y. & Seppänen, M. (2009). Tehdään elementeistä: suomalaisen betonielementtirakentamisen historia, SBK-säätiö, Helsinki, 332 p.

Kensek, K. & Noble, D. (2014). Building Information Modeling : BIM in Current and Future Practice (1), 1st ed. Wiley, Somerset, US, 432 p.

Lewis, B. (2016). Digitally designed – Qatar Faculty
of Islamic Studies, The Structural Engineer, Vol. 94(3), pp. 69-76.

Makris, M.P. (2013). Structural Design Tool for Performative Building Elements: A Semi-Automated Grasshopper Plugin for Design Decision Support of Complex Trusses, ProQuest Dissertations Publishing, Available:
http://search.proquest.com/docview/1459217046.

Mallintava suunnittelu, Betoniteollisuus Ry, web page. Available (accessed 16.1.2017):
http://www.elementtisuunnittelu.fi/fi/suunnitteluprosessi/mallintava-suunnittelu.

Piermarini, E., Nuttall, H., May, R. & Janssens, V. (2016). City of Dreams, Macau – making the vision viable, The Structural Engineer, Vol. 94(3), pp. 56-67.

Pro IT -tutkimus, Rakennusteollisuus, web page. Available (accessed 01.12.2017): http://virtual.vtt.fi/virtual/proj6/proit/.

Shepherd, P. (2011). Aviva Stadium – the use of parametric modelling in structural design, The Structural Engineer, Vol. 89(3), pp. 28-34.

Särkkä, E. (2015). Paalulaatan suunnittelu parametrisen tietomallin avulla, Tampereen ammattikorkeakoulu, Talonrakennustekniikka, Tampere, 37 p.

Tanska, T. & Österlund, T. (2014). Algoritmit puurakenteissa : menetelmät, mahdollisuudet ja tuotanto, B 32, 1st ed. DigiWoodLab, Oulun yliopisto, Arkkitehtuurin tiedekunta,

Tedeschi, A. (2011). Parametric architecture with Grasshopper: primer, 1st ed. Le Penseur, Brienza, Italy, 206 p.

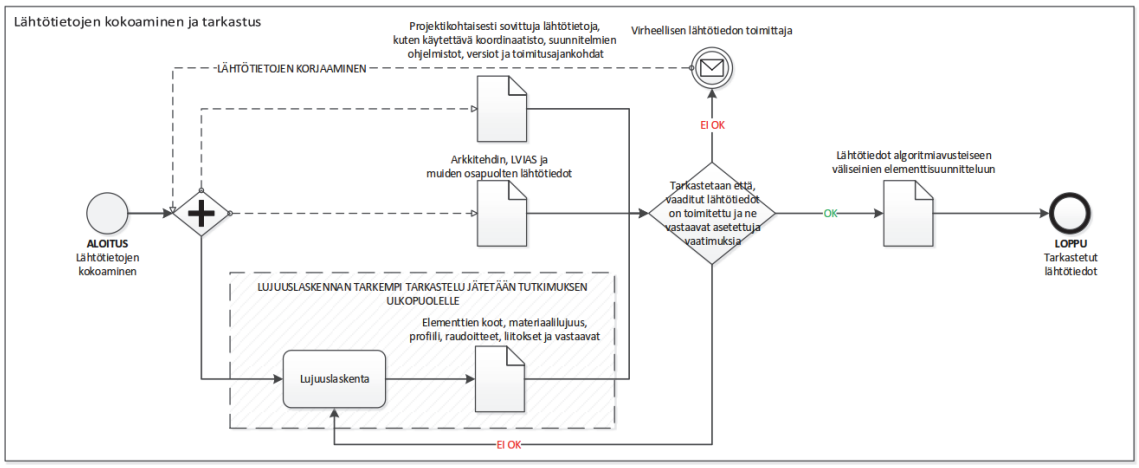
Trimble Solutions Corporation (2017). Tekla Structures Glossary, 161 p. Available: https://teklastructures.support.tekla.com/.

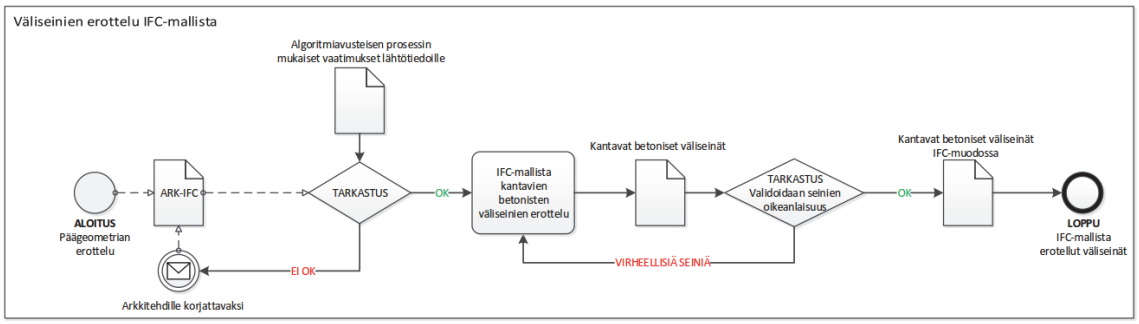
What are NURBS?, Robert McNeel & Associates, web page. Available (accessed 17.1.2017): https://www.rhino3d.com/nurbs.

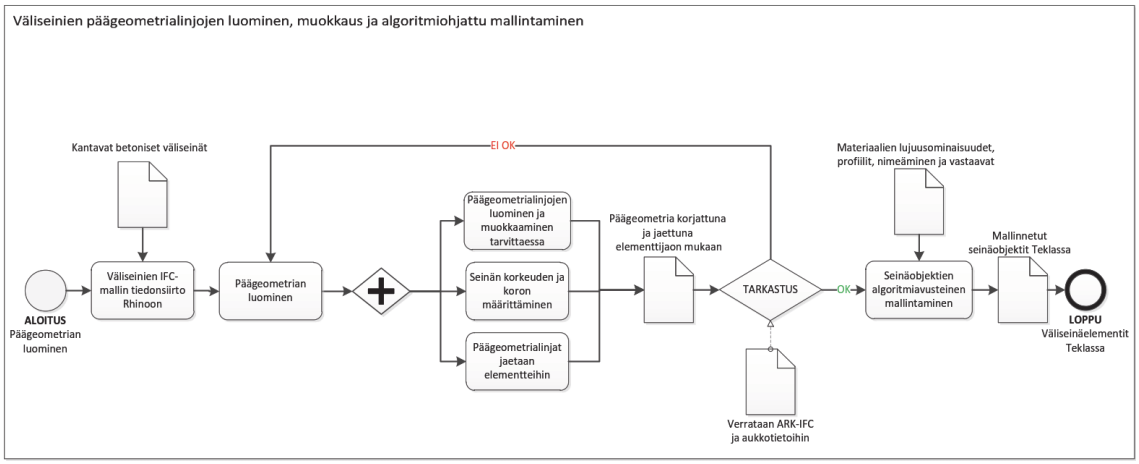
Woodbury, R. (2010). Elements of parametric design, Routledge, New York, USA, 300 p.

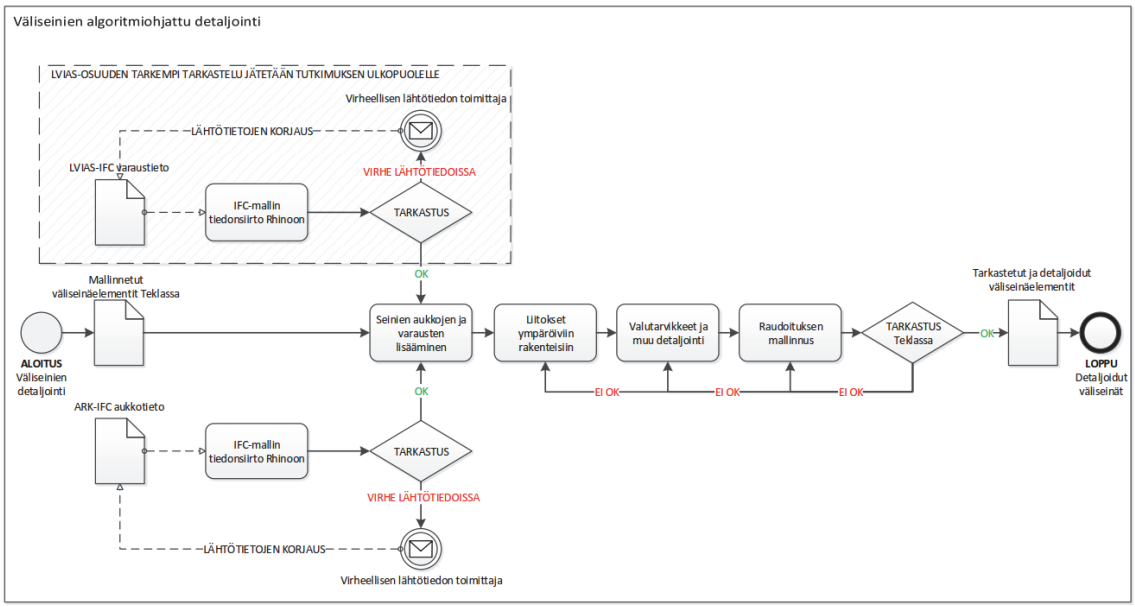
Yleiset Tietomallivaatimukset 2012, COBIM-hankkeen osapuolet, web page. Available (accessed 12.1.2017): http://buildingsmart.fi/yleiset-tietomallivaatimukset-ytv/.

LIITE A: VÄLISEINÄELEMENTIN ALGORITMIAVUSTEINEN-SUUNNITTTELUPROSESSIKAAVIO

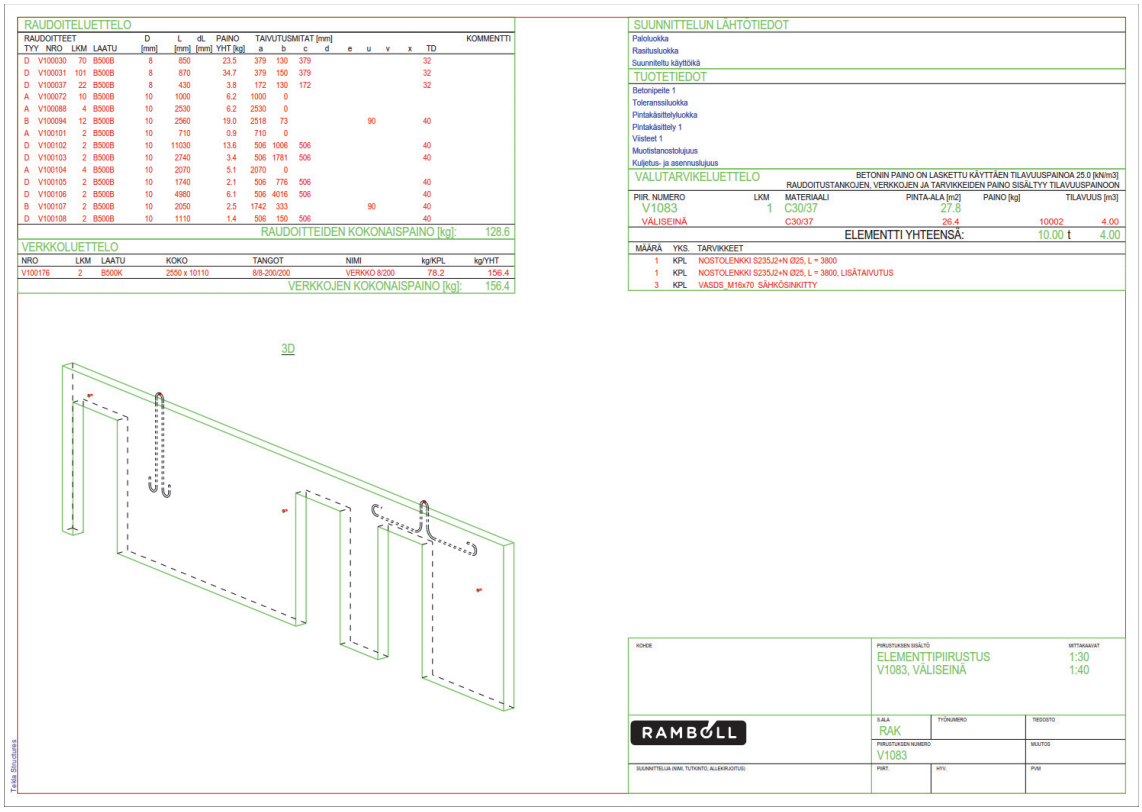







Väliseinien algoritmiohjattu detaljointi
LVIAS-OSUUDEN TARKEMPI TARKASTELU JÄTETÄÄN TUTKIMUKSEN ULKOPUOLELLE
Virheellisen lähtötiedon toimittaja
LÄHTÖTIETOJEN KORJAUS
LVIAS-IFC varaustieto
VIRHE LÄHTÖTIEDOISSA
IFC-mallin tiedonsiirto Rhinoon
TARKASTUS
OK
Mallinnetut väliseinäelementit Teklassa
Seinien aukkojen ja varausten lisääminen
Liitokset ympäröivin rakenteisiin
Valutarvikkeet ja muu detaljointi
Raudoituksen mallinnus
TARKASTUS Teklassa
OK
Tarkastetut ja detaljoidut väliseinäelementit
ALOITUS
Väliseinien detaljointi
LOPPU
Detaljoidut väliseinät
EI OK
EI OK
EI OK
OK
ARK-IFC aukkotieto
IFC-mallin tiedonsiirto Rhinoon
TARKASTUS
VIRHE LÄHTÖTIEDOISSA
LÄHTÖTIETOJEN KORJAUS
Virheellisen lähtötiedon toimittaja

LIITE B: VÄLISEINÄELEMENTIN VALMISTUSKUVA



RAUDOITELUETTELO

RAUDOITTEET TYY	NRO	LKM	LAATU	D [mm]	L [mm]	dL [mm]	PAINO YHT [kg]	TAIVUTUSMITAT [mm] a	b	c	d	e	u	v	x	TD	KOMMENTTI
D	V100030	70	B500B	8	850		23.5	379	130	379						32	
D	V100031	101	B500B	8	870		34.7	379	150	379						32	
D	V100037	22	B500B	8	430		3.8	172	130	172						32	
A	V100072	10	B500B	10	1000		6.2	1000	0								
A	V100088	4	B500B	10	2530		6.2	2530	0								
B	V100094	12	B500B	10	2560		19.0	2518	73					90		40	
A	V100101	2	B500B	10	710		0.9	710	0								
D	V100102	2	B500B	10	11030		13.6	506	1006	506						40	
D	V100103	2	B500B	10	2740		3.4	506	1781	506						40	
A	V100104	4	B500B	10	2070		5.1	2070	0								
D	V100105	2	B500B	10	1740		2.1	506	776	506						40	
D	V100106	2	B500B	10	4980		6.1	506	4016	506						40	
B	V100107	2	B500B	10	2050		2.5	1742	333					90		40	
D	V100108	2	B500B	10	1110		1.4	506	150	506						40	

RAUDOITTEIDEN KOKONAISPAINO [kg]: 128.6

VERKKOLUETTELO

NRO	LKM	LAATU	KOKO	TANGOT	NIMI	kg/KPL	kg/YHT
V100176	2	B500K	2550 x 10110	8/8-200/200	VERKKO 8/200	78.2	156.4

VERKKOJEN KOKONAISPAINO [kg]: 156.4

SUUNNITTELUN LÄHTÖTIEDOT

Paloluokka
Rasitustuokka
Suunniteltu käyttöikä

TUOTETIEDOT

Betonipeite 1
Toleranssiluokka
Pintakäsittelyluokka
Pintakäsittely 1
Viisteet 1
Muotistanostolujuus
Kuljetus- ja asennuslujuus

VALUTARVIKELUETTELO

BETONIN PAINO ON LASKETTU KÄYTTÄEN TILAVUUSPAINOA 25.0 [kN/m3]
RAUDOITUSTANKOJEN, VERKKOJEN JA TARVIKKEIDEN PAINO SISÄLTYY TILAVUUSPAINOON

PIIR. NUMERO	LKM	MATERIAALI	PINTA-ALA [m2]	PAINO [kg]	TILAVUUS [m3]
V1083	1	C30/37	27.8		
VÄLISEINÄ		C30/37	26.4	10002	4.00
			ELEMENTTI YHTEENSÄ:	10.00 t	4.00

MÄÄRÄ	YKS.	TARVIKKEET
1	KPL	NOSTOLENKKI S235J2+N Ø25, L = 3800
1	KPL	NOSTOLENKKI S235J2+N Ø25, L = 3800, LISÄTAIVUTUS
3	KPL	VASDS_M16x70_SÄHKÖSINKITTY

3D

KOHDE

PIIRUSTUKSEN SISÄLTÖ: ELEMENTTIPIIRUSTUS, V1083, VÄLISEINÄ

MITTAKAAVAT: 1:30, 1:40

RAMBØLL

SUUNNITTELIJA (NIMI, TUTKINTO, ALLEKIRJOITUS)

LAJA: RAK

TYÖNUMERO

TIEDOSTO

PIIRUSTUKSEN NUMERO: V1083

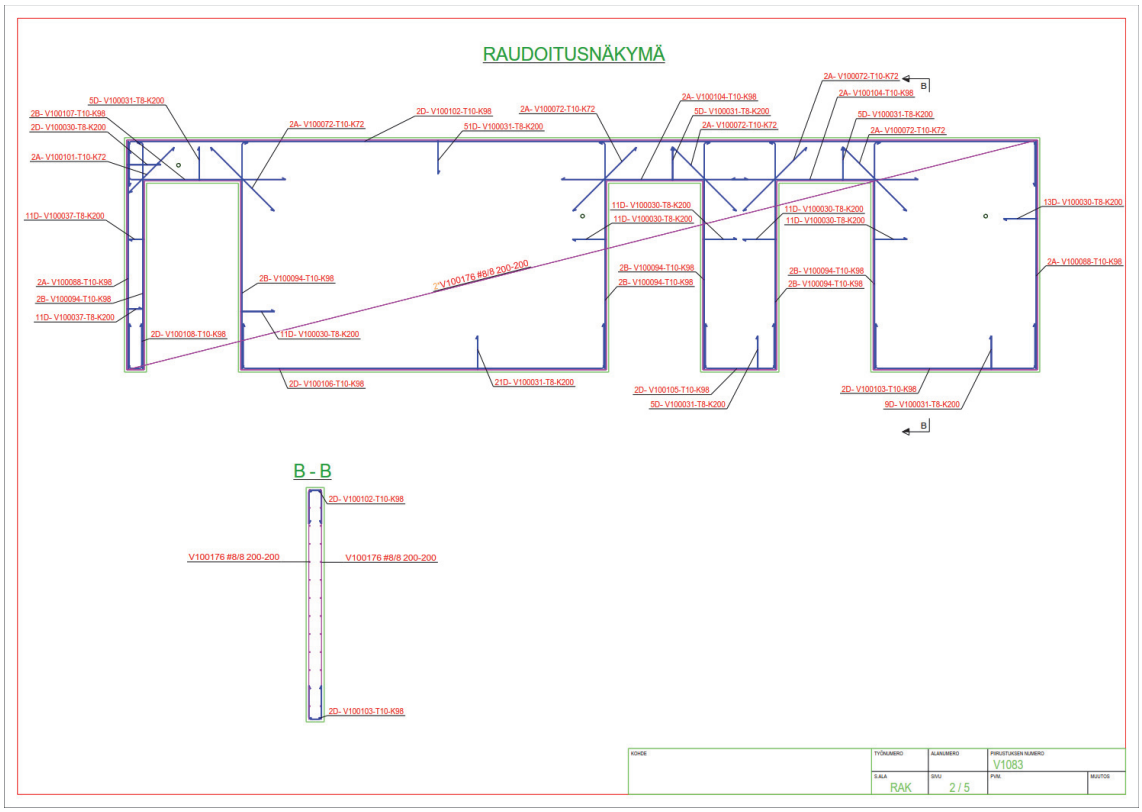
MUUTOS

PIIRT.

HYV.

PVM

Tekla Structures



RAUDOITUSNÄKYMÄ

5D- V100031-T8-K200
2B- V100107-T10-K98
2D- V100030-T8-K200
2A- V100101-T10-K72
11D- V100037-T8-K200
2A- V100088-T10-K98
2B- V100094-T10-K98
11D- V100037-T8-K200
2A- V100072-T10-K72
2D- V100102-T10-K98
51D- V100031-T8-K200
2B- V100094-T10-K98
2D- V100108-T10-K98
11D- V100030-T8-K200
2D- V100106-T10-K98
21D- V100031-T8-K200
2A- V100072-T10-K72
2A- V100104-T10-K98
5D- V100031-T8-K200
2A- V100072-T10-K72
11D- V100030-T8-K200
11D- V100030-T8-K200
2B- V100094-T10-K98
2B- V100094-T10-K98
2D- V100105-T10-K98
5D- V100031-T8-K200
2A- V100072-T10-K72
2A- V100104-T10-K98
5D- V100031-T8-K200
2A- V100072-T10-K72
11D- V100030-T8-K200
11D- V100030-T8-K200
2B- V100094-T10-K98
2B- V100094-T10-K98
2D- V100103-T10-K98
9D- V100031-T8-K200
13D- V100030-T8-K200
2A- V100088-T10-K98
2V100176 #8/8 200-200
B
B

B - B

2D- V100102-T10-K98
V100176 #8/8 200-200
V100176 #8/8 200-200
2D- V100103-T10-K98

KOHDE

TYÖNUMERO

ALANUMERO

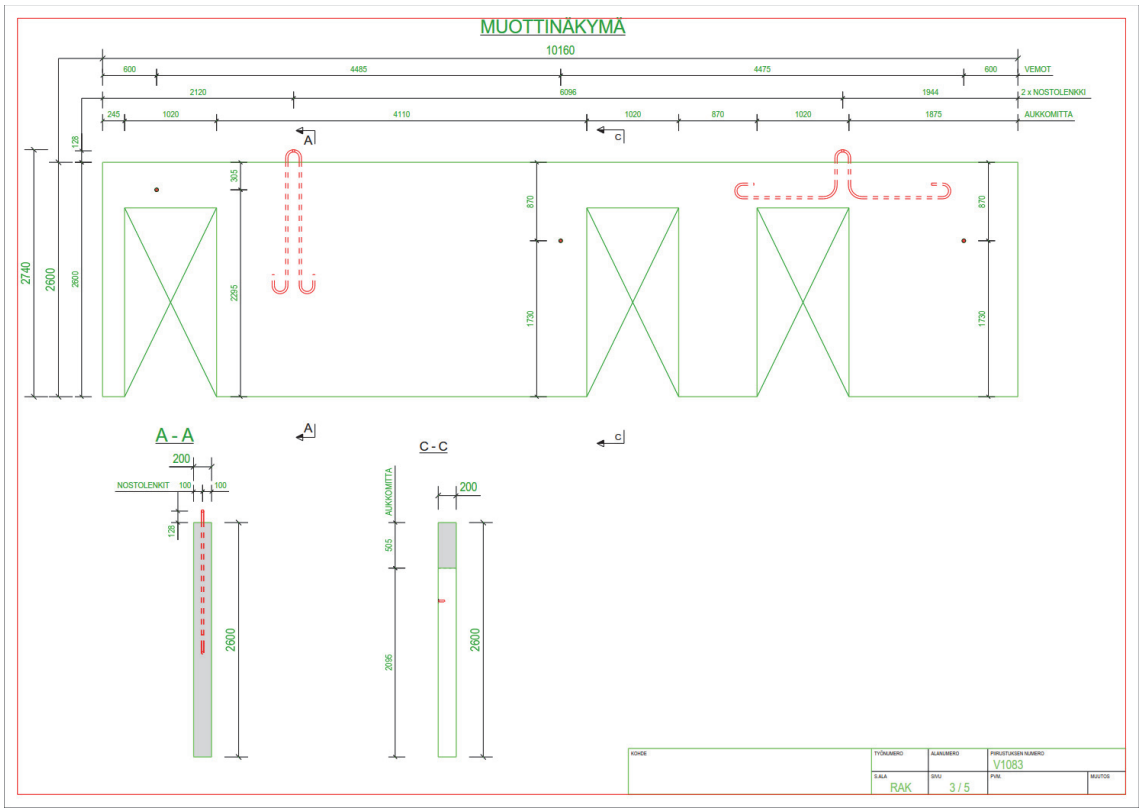
PIIRUSTUKSEN NUMERO: V1083

LAJA: RAK

SIVU: 2 / 5

PVM

MUUTOS


MUOTTINÄKYMÄ
10160
600
4485
4475
600
VEMOT
2120
6096
1944
2 x NOSTOLENKKI
245
1020
4110
1020
870
1020
1875
AUKKOMITTA
A
C
128
2740
2600
2600
305
2295
870
1730
870
1730
A - A
200
NOSTOLENKIT
100
100
128
2600
C - C
AUKKOMITTA
200
505
2095
2600
KOHDE
TYÖNUMERO
ALANUMERO
PIIRUSTUKSEN NUMERO
V1083
S-ALA
RAK
SIVU
3 / 5
PVM
MUUTOS

LIITE C: PROSESSIKAAVIO ELEMENTTISUUNNITTELUN RAKENNUSOSIEN ALGORITMIN LUOMISELLE